

Aalto University
School of Science
Degree Programme in Computer Science and Engineering

Géraud Le Falher

**FINDING SIMILAR NEIGHBORHOODS
ACROSS CITIES BY MINING HUMAN
URBAN ACTIVITY**

Where is Beverly Hills in your town?

Master's Thesis
Espoo, August 9, 2014

Supervisor: Professor Aristides Gionis
Advisor: Michael Mathioudakis, PhD

Aalto University
 School of Science
 Degree Programme in Computer Science and Engineering

ABSTRACT OF
 MASTER'S THESIS

Author:	Géraud Le Falher		
Title:	Finding similar neighborhoods across cities by mining human urban activity Where is Beverly Hills in your town?		
Date:	August 9, 2014	Pages:	58
Major:	Machine Learning and Data Mining	Code:	T-61
Supervisor:	Professor Aristides Gionis		
Advisor:	Michael Mathioudakis, PhD		
<p>We propose a method to match similar neighborhoods across different cities. That is, we give ourselves a measure of similarity between urban regions, as well as one region in one city. Our goal is then to find the region in some other cities which minimize the distance with the query region. Furthermore, we seek to do it efficiently, as it is prohibitive to evaluate the distance of all possible candidate regions.</p> <p>First, we collect trace of activities in 20 European and American cities from location aware social platforms Foursquare and Flickr. A thorough exploration of this dataset leads us to describe individual venues by relevant features including their aggregate activity across time, their visitors and overall popularity, and the typology of their surrounding. Then we learned several measures of venue similarity in a semi-supervised setting and evaluate their performance on two information retrieval tasks. After gathering human ground truth about neighborhoods, we evaluate different metrics between sets of venues and find out that Earth Mover’s Distance is best suited at assessing neighborhood similarity. Finally, we address the computational efficiency problem of finding the most similar neighborhood given a query. We devise a heuristic search strategy and show that it provides results of comparable quality while being orders of magnitude faster. This work has application in touristic recommendation and urban planning, as it provides a similarity measure between urban areas.</p>			
Keywords:	smart cities, metric learning, clustering, data mining, foursquare, flickr, earth movers distance, geolocation, neighborhood, twitter, urban computing		
Language:	English		

*I can feel it in my bones
So much left unknown
We continue to grow old
At least I found my gold*
— *Gold*, Vinyl Theathre¹

ACKNOWLEDGMENTS

I am greatly indebted to many people. First to my supervisor, Prof. Aristides Gionis, for suggesting me this topic, providing me thorough and kind guidance for more than six months, and for compensating my usual lack of enthusiasm with his. Coming second only by alphabetical order, I also want to offer my gratitude to my advisor, Dr. Michael Mathioudakis for his help and his fine remarks concerning my writing. This research was conducted while I was part of the Data Mining Group² and I am thankful for having met such nice and bright people (and for having eaten many different kinds of cake during the meetings). Actually, I may as well thank all the people I have worked with at Aalto University, because it was a real pleasure. Even more broadly, this time in Finland was a delight, made possible by the impressive, pristine nature as much by the kindness of people living there.

In addition to these talented and supportive people, this work would not have been possible if not for many great open-source softwares. Yet because it would feel weird to thank them in the same place, I relegate them in [Appendix B](#).

On a personal level, I would first like to thank the members of my family for their unconditional support. I have also heavily relied on friends for their knowledge on Paris; yet neither Camille Autran, Oliver Braud nor Aloïs Guillopé have ever expressed any irritation (at least openly). As I am finishing the redaction of this manuscript, I have to acknowledge that, however frequent and unexpected, the interruptions of Kiran Garimella were probably beneficial in the long term. Finally, the outcome would certainly have been different without the influence of Sanja Šćepanović; both objectively—for giving me the Twitter API key used to collect my dataset— and subjectively—for so many reasons that it would be pointless to try to condense them in a few sentences.

¹ From their album of the same name: youtu.be/y_hSArYE4kI.

² research.ics.aalto.fi/dmg/

CONTENTS

1	INTRODUCTION	1
1.1	Motivation	1
1.2	Outline	2
2	PROBLEM DEFINITION AND NOTATION	3
2.1	Cities, venues and activities	3
2.2	Distance between venues	3
2.3	City neighborhoods	4
2.4	Distance between neighborhoods	4
3	MEASURING SIMILARITY	6
3.1	Clustering	6
3.1.1	Dimensionality reduction	7
3.1.2	Spatial data structure	8
3.2	Metric	9
3.2.1	Ground metric	9
3.2.2	Set metric	11
4	DATASETS	13
4.1	Foursquare	13
4.1.1	Check ins	14
4.1.2	Venues	14
4.2	Flickr	14
4.3	Exploration	16
4.3.1	Time	16
4.3.2	Space	20
4.3.3	Location entropy	20
4.3.4	Photos	21
4.4	Chosen representation	23
4.5	Human input	23
4.6	Limitations and other sources	23
5	CHOICE OF METRICS	28
5.1	Venues metrics	28
5.1.1	Brand finding	29
5.1.2	Category ordering	30
5.2	Neighborhoods metrics	31
5.2.1	Candidates	31
5.2.2	Evaluation	33
6	IMPROVING SEARCH EFFICIENCY	36
6.1	Method	36
6.2	Results	39
6.2.1	How well it approximates exhaustive EMD	39
6.2.2	How well it matches ground truth	40

7	RELATED WORK	43
8	DISCUSSION	46
8.1	Other applications	46
8.2	Conclusion	46
8.3	Future work	47
A	DATASET FORMAT	49
B	MORE ACKNOWLEDGMENTS	51
	BIBLIOGRAPHY	52

LIST OF FIGURES

Figure 1	Neighborhood of a point, from [WSog].	10
Figure 2	Check-ins temporal pattern.	17
Figure 3	Venues clustered by time of check-ins	18
Figure 4	Venues cluster by time among all the cities	19
Figure 5	Venue density in Paris	20
Figure 6	Two different measure of entropy	22
Figure 7	Website survey interface.	24
Figure 8	2D projection of European venues by t-SNE	29
Figure 9	Ranking of target city venues by distance to query venus.	38
Figure 10	Recall vs coverage of k nearest neighbors	38
Figure 11	Approximation trade off with respect to ℓ	41
Figure 12	Approximation trade off with respect to k	42

LIST OF TABLES

Table 1	Dataset number	15
Table 2	Venue features	26
Table 3	Ground truth neighborhoods	27
Table 4	Metric score for brand task	30
Table 5	Metric score for brand task	31
Table 6	Metric scores for category task	32
Table 7	Average score of each metric	35
Table 8	Approximation ability to recover ground truth	40
Table 9	Check-in format	49
Table 10	Photo format	49
Table 11	Venue format	50

INTRODUCTION

Give a high level overview of the problem. But why should we solve this problem? We provide some answers related to touristic recommendations and urban planning in [Section 1.1](#), before giving a short overview of the thesis in [Section 1.2](#).

1.1 MOTIVATION

More and more people are living in cities¹. This simple observation leads to two questions: how to help urban dwellers make informed everyday decisions and how to understand these large and complex systems. Fortunately, now is a good time to tackle these issues. Indeed, thanks to social networks and mobile devices, we know where and when people are active within cities [\[12a\]](#). We seize this opportunity to devise a similarity measure between urban areas. First we motivate the need for such a measure through a typical use case:

Let us say you have lived in a city for the last years. During this time, you have acquired urban knowledge that you can leverage to find relevant locations where to perform various tasks. In simpler words, whether you want to buy milk, listen to loud music or exhibit your athletic skills, you know where to go. Yet life is full of surprises and you may be spending a few days in a new town. Fair enough, but you do not want to renounce to your wide range of outside activities. This is where our application comes in handy. You select a location or an area in your familiar city and it shows you a place or a zone sharing similar characteristics in your surroundings. Thus you can boldly venture into distant shores, reassured that a part of your home will always travel with you.

As this short illustration demonstrates, our problem has applications to recommending locations in a city. But it is also applicable in the analysis of cities and urban planning. For instance, when applied to the neighborhoods of one city, our techniques allow to identify neighborhoods that are similar to each other, and thus can help us understand what is going on in each area, what are the hubs of different activities, how citizens are experiencing the city, and how they are utilizing its resources.

¹ According to an [UN report \[12b\]](#), “the population living in urban areas is projected to [pass] from 3.6 billion in 2011 to 6.3 billion in 2050.”

1.2 OUTLINE

Here we give an outline of the thesis structure. After exposing our problem in details in [Chapter 2](#), the rest of this work will be devoted to present which solutions we used to solve it and how well they perform. We start in [Chapter 3](#) by introducing some common techniques concerning similarity measure that serve as building blocks for our own methods. Namely, we talk about clustering, how it relates with dimensionality reduction and say a few words about data structures for efficient implementation. Then we discuss learning metrics between single object and introduce the metrics we used to compare sets of such object. [Chapter 4](#) is dedicated to the datasets on which we conducted our experiments. We describe how we collected data from 20 cities in Europe and in the US; namely activity logs from Foursquare and Flickr, a [Location Based Social Network \(LBSN\)](#) and a photo-sharing platform respectively. We explore the organization of these activities in time, in space, and with respect to various measures of entropy, as well as the interplay between the two sources. This let us choose a set of numeric features which accurately describes the characteristics and the overall activity of locations inside cities.

These careful preparations finished, we start presenting results about metrics in [Chapter 5](#). First we devised two information retrieval tasks based on Foursquare categories to assess the ability of several metrics to match venues together. We show a slight advantage of metrics whose parameters are learned from the data compared with simple Euclidean distance. We remark as well that accuracy is generally low because information as venue level is rather noisy, which further motivate the study of neighborhood similarity. Using ground truth data obtained from a user study conducted over Internet, we compare several metrics and show that *Earth Mover's Distance* is the most able to agree with human perception.

To select EMD, we perform an exhaustive search, which is rather time consuming. Therefore, in [Chapter 6](#), we describe the design of a faster search strategy. It relies on a pre-processing of venues in the target city with respect to the query in order to prune the search space. The remaining venues are then clustered in space to form a few candidate neighborhoods. We illustrate experimentally that the speedups thus obtained come at little accuracy cost and in some cases, even provide better results. Finally, we give a list of references to related works in [Chapter 7](#) and we conclude in [Chapter 8](#) by discussing other setting where our method can be applied and offering some directions for future work.

PROBLEM DEFINITION AND NOTATION

Here we introduce some notation and describe our data model before defining formally the two problems we want to solve: first, learn an appropriate metric between neighborhoods and second, use it efficiently during the search process.

2.1 CITIES, VENUES AND ACTIVITIES

We consider a set \mathcal{C} of n cities, $\mathcal{C} = \{C_1, C_2, \dots, C_n\}$. Each city C contains a set of *venues* $V(C)$. A venue is an uniquely identified location that can be visited by individuals. It includes for example restaurants, shops, airports, monuments and parks. We denote the set of all venues by $\mathcal{V} = \bigcup_{C \in \mathcal{C}} V(C)$.

Each city C is also described by another subset $A_C \subseteq \mathcal{A}$, where \mathcal{A} is a set of *activities*. These activities are user contributed discrete points in space and time that can hold additional attributes. Here, we exploit Flickr photos (with tags) and Foursquare check-ins as type of activity.

In general, many other types of activity can be considered. Geo-located tweets fit our model directly, as a tweet is composed by a user, has spatio-temporal coordinates, and contains text. The latter may conveys sentiment, which can be extracted via sentiment-analysis methods [PLo8]. Reviews collected from Yelp or other reviewing platforms also fit our model nicely. We could also consider noise and pollution measurements or traffic flow for cars, pedestrians, and bikes, measured by city sensors or mobile devices. Stretching the concept, one could even consider collecting anonymized customer receipts (e.g., grocery shops, restaurants, or other services) and record the distribution of services provided and goods consumed in a city.

Activities are associated with venues, either explicitly, in the case of check-ins, or implicitly, as with photos. In the latter case, the relationship between venues and activity is based on spatial distance. By combining venues own characteristics and their associated items, we represent them by a *feature vector* $\mathbf{f}(v)$ (see Section 4.4 on page 23 for a complete description).

2.2 DISTANCE BETWEEN VENUES

Given two venues $v_i, v_j \in \mathcal{V}$ with feature vectors $\mathbf{f}_i = \mathbf{f}(v_i)$ and $\mathbf{f}_j = \mathbf{f}(v_j)$, the simplest distance one can define between the two venues is the p -norm between their feature vectors $\|\mathbf{f}_i - \mathbf{f}_j\|_p$. This simple definition has a number of shortcomings, as it does not account for (i) the different scale of different features, (ii) the different scale of the same feature due to variations in the cities, (iii) the importance of features, and (iv)

potential correlations between features. One way to address the first two shortcomings is to ensure that each feature, aggregated at a city level, has mean equal to zero and standard deviation equal to one.

To address the other two shortcomings, we can introduce a parametrization θ of usual distances. Learning this parameter θ , and thus the distance measure between venues, is the first challenge we face:

Objective 1 *Learn the parameter θ so that the distance measure d_θ captures best the human perception of similarity between venues.*

The theoretical motivation for metric learning is exposed in [Section 3.2.1](#) and our own results are described in [Section 5.1](#).

2.3 CITY NEIGHBORHOODS

Summarizing our discussion so far, each venue $v \in \mathcal{V}$ is described by the pair $(\text{loc}(v), \mathbf{f}(v))$, where $\text{loc}(v)$ and $\mathbf{f}(v)$ specify the location of the venue v , and its feature vector, respectively.

Now, given a city $C \in \mathcal{C}$, we define a *neighborhood* (or a *region*) R of the city C as a geographical region (a closed and connected set in the geographical plane). We abuse notation and we write $R \subseteq C$ to denote that R is a neighborhood of the city C . For a neighborhood $R \subseteq C$ we define $V(R)$ to be the set of venues of C that are contained in R , i.e.,

$$V(R) = \{v \in \mathcal{V} \mid \text{loc}(v) \in R\}.$$

We also define $F(R)$ to be the set of feature vectors of all the venues in R , that is,

$$F(R) = \{\mathbf{f}(v) \mid \text{loc}(v) \in R\}.$$

2.4 DISTANCE BETWEEN NEIGHBORHOODS

Our objective is to define a meaningful distance measure between city neighborhoods. We want two neighborhoods to be similar, if they host comparable activities. Thus, given two neighborhoods R_i and R_j we consider the feature vectors $F(R_i)$ and $F(R_j)$ of the venues contained in the two neighborhoods, and we define the distance $\delta(R_i, R_j)$ between R_i and R_j by

$$\delta(R_i, R_j) = D(F(R_i), F(R_j)), \tag{1}$$

where D is a distance function between *sets of feature vectors*. Our second objective is stated as follows.

Objective 2 *Design a distance measure δ between city neighborhoods as expressed by (1), i. e. a distance measure D between sets of feature vectors of the venues in the two neighborhoods. The distance measure should capture best the human perception of similarity between city neighborhoods.*

In [Section 5.2](#) we consider a number of different options for the distance function D , and we discuss our methodology for selecting the best one. Building on the optimal distance measure selected for our objective, we then consider the following *city-neighborhood search* problem.

Problem 1 *We are given a neighborhood R in a city $C \in \mathcal{C}$ and a subset of target cities $\mathcal{C}' \subseteq \mathcal{C}$. The goal is to find a neighborhood R' in some city $C' \in \mathcal{C}'$ so that the distance $\delta(R, R')$ is minimized.*

Two interesting special cases of Problem 1 are (i) $\mathcal{C}' = \mathcal{C}$, search for the most similar neighborhood in all cities; and (ii) $\mathcal{C}' = \{C'\}$, search for the most similar neighborhood in a given city C' . The emphasis for Problem 1 is on computational efficiency, since given a distance function δ , one can always apply a brute-force search. We discuss our solution in [Chapter 6](#).

3

MEASURING SIMILARITY

In this chapter, we provide an overview of the theoretical background of many methods we employed to solve our problem. Because it does not contribute directly to our solution, it can be skipped by readers already familiar with clustering and metric learning. Yet it introduces many concepts on which we rely later. As “finding similar neighborhoods” entails, the main theme is measuring similarity between objects. This can be used to cluster them in meaningful groups (Section 3.1). An important step is to tailor general metric toward our goal (Section 3.2). Indeed, Balcan, Blum, et al. [BBS08] show that “good” properties of similarity functions are transferred to classifiers using them.

3.1 CLUSTERING

A general class of methods addressing unsupervised problems is clustering. “*Clustering algorithms partition a given data set into several groups based on some notion of similarity between objects*” [LBB05]. The most widely used is the ***k*-means algorithm** [Mac67]. After initially choosing k centroids, an iterative process takes place, divided in two phases. First each point is assigned to the cluster of the closest centroid. Then centroids positions μ_i are updated by taking the mean of all points belonging to their cluster. Convergence occurs when centroids positions do not change between iterations. In practice, the algorithm is fast but it is only guaranteed to find a local optimal of the within-cluster sum of squares:

$$\sum_{i=1}^k \sum_{x_j \in \text{cluster}_i} \|x_j - \mu_i\|^2$$

Another drawback is that k , the number of clusters, has to be specified before algorithm execution, whereas this information is often unknown at this stage. Finally, because it results in a Voronoy diagram, the clusters found are linearly separable, which may not reflect the actual data.

Because of these limitations, numerous alternatives have been developed. A cluster can be defined as an area of high density surrounded by low density regions. This definition suggests assigning a numeric value of density to every point of space. A common method is **Kernel Density Estimation** [Ros56]. A kernel, generally parametrized by its bandwidth h , is centered around each point (i.e. a symmetrical weight-

ing function, for instance a Multivariate Gaussian) and the estimation of the probability distribution f is their normalized sum:

$$\hat{f}_{\mathbf{h}}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n K_{\mathbf{h}}(\mathbf{x} - \mathbf{x}_i)$$

Although it is a useful information by itself, this density estimation does not directly provide a clustering. One idea would be to find the modes of \hat{f} , which is the underlying principle of Mean Shift [Che95]. Another density based algorithm is **DBSCAN** [Est+96]. It has two parameters: a distance ϵ and a number of points minPts. A core sample is a point with at least minPts neighbors in its ϵ -neighborhood. From a core sample, its neighbors are visited to find other core samples belonging to the same cluster. Once it is not more possible, the algorithm moves to another point. Points that are ϵ close to a core sample without having a large enough neighborhood are also part of the cluster (but are deemed fringe points rather than core ones). Any other remaining point is noise. With an appropriate index structure (see Section 3.1.2), neighborhood queries takes $O(\log n)$ time and the algorithm runs in $O(n \log n)$. Otherwise, one need to compute the pairwise distance matrix, which takes $O(n^2)$ time and memory. DBSCAN finds an arbitrary number of clusters with arbitrary shapes. In the special case where points carry user identification (like photos or tweets), it can be tweaked to favor areas that exhibit large user diversity [KMK10]. Another extension, OPTICS, can identify clusters despite data having varying density, but it only produces a reachability-plot, which requires further processing to be transformed into clusters [Ank+99].

A major component of all clustering algorithm is the distance function, and the space where it operates. The default choice is the Euclidean metric in the original feature space but these two parameters can be changed. In the latter case, it often leads to dimensionality reduction, whereas the former suggests metric learning, which will be discussed in Section 3.2.

3.1.1 Dimensionality reduction

Many methods are based on the idea of preserving distances between points, under the general name Multidimensional Scaling [De 77]. Given points i and j in the original space, we know their separation $\delta_{i,j}$ with a weight $w_{i,j}$ ¹. We are looking for their new position x_i and x_j in the reduced space such that the stress

$$S = \sum_{i,j} \sqrt{\frac{\sum w_{i,j}(\delta_{i,j} - d(y_i, y_j))^2}{\sum d(y_i, y_j)^2}}$$

is minimized.

¹ This weight denotes either confidence in the measurement or importance of the pair of points.

Instead of computing distances in a geometrical sense, it is also possible to give them a probabilistic interpretation. Namely, in Stochastic Neighbor Embedding [HR02], starting from the dissimilarity between two points i and j , $d_{ij}^2 = \frac{\|x_i - x_j\|^2}{2\sigma_i^2}$, we compute the probability of i picking j as a neighbor in the original space

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik}^2)}$$

We do the same in the low dimensional space, where positions are denoted by y_i

$$q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

and we set the y_i to minimize $\sum_i \text{KL}(P_i \| Q_i)^2$, which can be understood as preserving the neighborhood of each point. Later was introduced t -SNE [MH08], which makes the optimization problem simpler and avoid the crowding problem by modeling distances in the low dimensional space with a heavy tail Student t -distribution with one degree of freedom

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq i} \left(1 + \|y_i - y_k\|^2\right)^{-1}}$$

An ingenious implementation runs in $O(n \log n)$ [Maa13].

It is easier to visually evaluate clustering results in this two or three dimensional space, although dimensions are not necessarily meaningful. Furthermore, we assume that sample points are not randomly distributed in the whole space. Thus, we are trying to recover the low dimensional manifold where they lie.

3.1.2 Spatial data structure

A common subproblem of many methods exposed so far³ is to retrieve the k closest neighbors of a given point. If there are n candidates in a d dimensional space, a naive linear scan takes a prohibitive $O(n \cdot d)$ time. Fortunately, there are data structure well suited to solve this problem faster, even though they become less efficient as d increases. Namely, we will describe k -d tree and ball tree. The common idea behind them is to partition space according to points distribution.

A k -d tree [Ben75] is a binary search tree whose every node is a hyperplane that separates the space \mathbb{R}^k in two parts, and whose leaves contain points lying in the region of the space define by all the parents hyperplanes. These hyperplanes are aligned with axes, meaning that

² Here, KL is the Kullback–Leibler divergence, which will presented shortly.

³ This subproblem is also interesting by itself.

on the left of a node at the i^{th} level are all points whose coordinate i is smaller than t and the right the points for which is larger. This threshold t is chosen as the median value of this coordinate (or at least a good approximation of it) to keep the tree balanced. The tree is constructed recursively until there are not enough points at a given sub level to justify further splitting. After that, the main operation is retrieving the point who is the closest to a given input point. By looking at its coordinates, we can go down the tree in $O(\log n)$ time to restrict the number of points to look for.

Ball trees [Omo89] are also binary trees but this time, nodes are hyperspheres of \mathbb{R}^k , or balls. The ball at each node is the smallest ball that contains all the balls in the corresponding subtree, and leaves are balls which enclosed some points of the dataset. Therefore, for a ball S (of center c and radius r) and a query point q , we know that if $q \notin S$, q cannot be closer to points in S than $\|c - q\| - r$. This fact is used at query time to prune parts of the tree.

Another data structure of interest is the R-tree [Gut84], which is able to store and retrieve non-zero size geometry such as polygon. There, nodes are hyperrectangles and leaves contain only one object. It is again a balanced search tree, but not a binary one. Each node can store up to M entries. The idea behind search is the same, taking advantage of the nested box to prune part of the tree. The main difficulties lie in the efficient construction of the tree as well as fast subsequent insertions.

3.2 METRIC

3.2.1 Ground metric

As we mentioned, to improve performance of classifiers who rely on distances⁴, we must change the way distances are measured. Either by computing them in a new space, as in dimensionality reduction, or by directly modifying the underlying norm, that is replacing the L_2 norm $\|x - y\|_2^2$; for instance by the L_1 norm (Manhattan distance) or the L_∞ . Noting that $\|x - y\|_2^2 = (x - y)^T A(x - y) = d_A(x, y)$, where $A = I$, one can also replace A with any positive definite matrix to still define a metric. A well-motivated choice is setting A as the inverse of the covariance matrix, corresponding to the Mahalanobis distance [Mah36]. But A can be learned from the data in other semi-supervised ways, by specifying constraints between pairs of points. These constraints are based on class labels, meaning we want points of the same class to be close and points of different classes to be set apart. In this work, we consider two methods that optimize A subject to such pairwise constraints.

The first is **Information Theoretic Metric Learning** [Dav+07]. Given a positive definite matrix A , we can associate it with a multivari-

⁴ The textbook example being k -nearest neighbors classifier.

ate Gaussian distribution of mean μ and covariance A^{-1} (uniquely, up to a scaling constant):

$$p(x; A) = \frac{1}{Z} \exp \left(-\frac{1}{2} d_A(x, \mu) \right)$$

We also need a reference matrix A_0 (for instance the identity or the covariance matrix of the dataset), a set S of similar points and a set D of dissimilar points. We want A to be close to A_0 in the sense of relative entropy $\text{KL}(p(x; A_0) || p(x; A)) = \int p(x; A_0) \log \frac{p(x; A_0)}{p(x; A)} dx$ by solving

$$\begin{aligned} \min_A \quad & \text{KL}(p(x; A_0) || p(x; A)) \\ \text{subject to} \quad & d_A(x_i, x_j) \leq u & (i, j) \in S, \\ & d_A(x_i, x_j) \geq l & (i, j) \in D. \end{aligned}$$

The other one is **Large Margin Nearest Neighbor** [WS09] Specif-

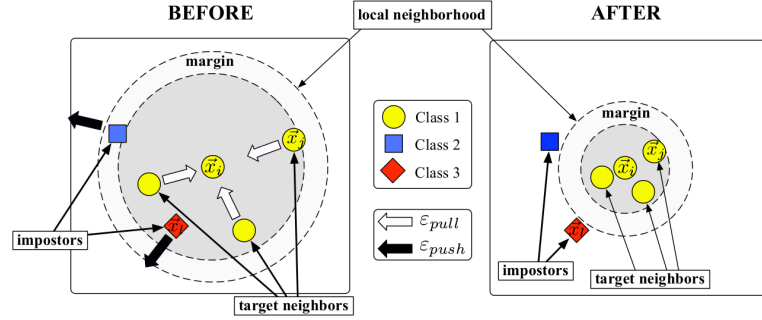


Figure 1: Neighborhood of a point, from [WS09].

ically, let N_i be the set of the k closest neighbors of x_i in the original space which share the same class, and called them target neighbors. Closer points of different class are impostors (see Figure 1). The optimization pushes target neighbors close to x_i while pulling away impostors. The number of active constraints is linear rather than quadratic because only the neighborhood of each points contributes to it. Thus the following semidefinite program can be solved efficiently:

$$\min_A \sum_{i,j \in N_i} \underbrace{d_A(x_i, x_j)}_{\text{pull target neighbor } x_j \text{ closer}} + \mu \underbrace{\sum_{k, y_k \neq y_i} [1 + d_A(x_i, x_j) - d_A(x_i, x_k)]_+}_{\text{push impostor } x_k \text{ beyond target neighbor } x_j}$$

where $[x]_+$ is the hinge-loss: $[x]_+ = \max(0, x)$

The end result is a linear, global metric but there are other approaches. One can learn multiple local metrics, or a non linear metric like **Gradient Boosting LMNN** [Ked+12]. It solves a similar optimization problem except that distances are computed in a new space, obtained by a non linear additive mapping. This mapping is itself

learned by combining multiple regression trees of limited depth selected by gradient boosting.

A more complete overview of metric learning is given in the survey of Bellet, Habrard, et al. [BHS13].

3.2.2 Set metric

We discussed metrics between points but in this work, we also used distances between sets of such points. If we interpreting their coordinates as geometric positions in \mathbb{R}^k , there are some easy to compute metrics. It includes, for instance, the bottleneck distance [EI96] (the smallest distance between pairs of farthest points) and the Hausdorff distance (the largest distance between pairs of closest points) along with its variations [DJ94].

But such approaches are rather sensible to outliers and do not perform so well in high dimension, as distances tend to be more uniformly distributed there. Therefore, we want to involve all the points from both sets in the computation. One way to do this is to build a bipartite graph with edges weighted by the opposite distance between points and find a maximum weight matching. This can be solved by the Hungarian algorithm [Mun57] in $O(n^4)$, a precursor of primal-dual method to solve linear program. This bound was later improved to $O(n^3)$ by considering it as a flow problem [EK72].

Because this problem has a min cost flow constraints matrix, it is are totally unimodular (i. e. every non singular sub square matrix has a determinant of +1 or -1) and thus admits an integer solution. Yet it may be more convenient to consider non integer supply and demand, which leads to fractional weight and probabilistic interpretation. This is then called the transportation problem and the modern Kantorovitch formulation makes the probabilistic aspect more apparent:

$$\inf_{\gamma \in \Gamma(\mu, \nu)} \int_{X \times Y} c(x, y) d\gamma(x, y)$$

Here X and Y are two metric spaces, c a measurable function between them which represents the cost of moving from different parts of X to some parts of Y . We look for a measure γ whose marginals on X and Y are μ and ν (the set $\Gamma(\mu, \nu)$ is called the set of all *couplings* of μ and ν).

It is closely related to the Wasserstein distance of two probability measures μ and ν of a space M with p^{th} finite moments

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu, \nu)} \int_{M \times M} c(x, y)^p d\gamma(x, y) \right)^{\frac{1}{p}}$$

W_1 is also called **Earth Mover's Distance** [RTG98]. Intuitively, it measures the *total amount of work* needed to transform (move) one set

of positions (total mass) to the other. It can also be defined as the minimal cost of the following linear program

$$\begin{aligned} & \min_f \sum_{i,j} d_{i,j} f_{i,j} \\ \text{subject to } & \sum_j f_{i,j} \leq w_{P,i} \\ & \sum_i f_{i,j} \leq w_{Q,j} \\ & \sum_{i,j} f_{i,j} = \sum_i w_{P,i} = \sum_j w_{Q,j} = 1 \end{aligned}$$

where $d_{i,j}$ is the distance between the i^{th} point of P and the j^{th} point of Q , and w_P and w_Q are the weights of these points, summing up to 1.

As we used EMD extensively, it worth mentioning that since it was introduced, there have been efforts to improve its performance while preserving its characteristics. For instance, Ling and Okada [LO07] present a lower complexity algorithm that uses L_1 as the ground metric. Shirdhonkar and Jacobs [SJ08] propose another approximation algorithm based on wavelet decomposition but the complexity is exponential with the number of dimension, which makes it intractable when d is greater than 6. Pele and Werman [PW09] show that assuming an upper bound over all pairwise distances, one can further simplify the optimization formulation, which improves running time and better match the human perception of colors in computer vision. Finally, the Sinkhorn distance [Cut13] is closely related to optimal transport and can be used to quickly obtain lower and upper bounds of EMD, especially when run over GPUs. But it also performs well on classification tasks on its own.

As hinted in EMD by the link between distance in Euclidean spaces and probability spaces, we can also measure similarity between sets of objects if we describe them by probability distributions. Information theory provides us with multiple tools for assessing distance between distributions, the most well known being the Kullback–Leibler divergence, defined in the discrete case as

$$\text{KL}(p||q) = \sum_i p_i \log \left(\frac{p_i}{q_i} \right)$$

However, this divergence is not a metric as it is not symmetric (nor does it satisfies the triangular inequality). Thus, we used a metric based on it, the **Jensen–Shannon Divergence** [ES03], which is defined using the average of the two distributions $m = \frac{p+q}{2}$

$$\text{JSD}(p; q) = \frac{1}{2} (\text{KL}(p||m) + \text{KL}(q||m))$$

To solve the problem at hand, we extract information from data. Specifically, we take advantage of geo localized activities exposed on Foursquare (Section 4.1) and Flickr (Section 4.2). An extensive exploration of the data collected (Section 4.3) leads us to choose *venue* as the elementary unit of information. Therefore, we represent them by some numerical features (Section 4.4) in the rest of this work. Beside raw data, we also gathered neighborhoods ground truth by conducting a small-scale user study (Section 4.5). Finally, as we are problem and not data-driven, we conclude this chapter by assessing the adequacy of such data toward our goal, and by presenting alternative sources of information (Section 4.6).

To limit the scope of our study, we focus on 20 cities:

- 10 located in the United States: Atlanta, Chicago, Houston, Indianapolis, Los Angeles, New York, Saint Louis, San Francisco, Seattle and Washington;
- 10 located in Europe: Amsterdam, Barcelona, Berlin, Helsinki, London, Moscow, Paris, Prague, Rome and Stockholm.

They were chosen for their high activity according to 4sqstat.com. We deliberately exclude other parts of the world as we are not familiar enough with them to correctly evaluate our results.

At the time of writing, this dataset is available online¹.

4.1 FOURSQUARE

Foursquare is the most popular [LBSN](#) in 2014, claiming more than 50 million users². Its two main purposes are: (i) enable users discover new places, and (ii) let their friends or the world know where they are. The first goal is achieved through venues. Venues have their own web page that displays basic facts (like name, address and type) but also user contributed information (such as photos, reviews, likes and tips) which can be used for rating or recommendation. The second goal revolved around check-in, which represents the visit of a user to a venue at a given time. It is also associated with a web page, potentially displaying a message or a photo enclosed with the check-in.

¹ figshare.com/authors/Géraud_Le_Falher/542931

² See foursquare.com/about for more usage statistics.

4.1.1 *Check ins*

Following Foursquare’s privacy policy, check-ins are private. Yet some users opt to share their check-ins via Twitter³ a popular micro-blogging platform. As Twitter allows anyone to access a 1% sample of public tweets⁴, we collected such check-ins from March to July 2014 by monitoring tweets with **4sq.com** links and retaining those located in the target cities. Essential informations of these 1,669,878 particular tweets are user id, Foursquare venue, local time, latitude and longitude. Yet they contain other fields, whose details are given in [Table 9](#) on page 49.

In addition, we used a previously released dataset [Che+11], extracting 1,296,505 Foursquare check-ins generated between September 2010 and January 2011 in these cities⁵.

Finally, we obtained more data by crawling the timeline of users that appear to post geo located tweets according to the public sample. We apply this approach in Barcelona, Helsinki, Paris, Rome, San Francisco, Stockholm and Washington, resulting in 2,087,227 additional tweets (even though a significant portion of them were located in other cities because some users travel or where simply visiting the city).

In total, our Foursquare data consists of more than 5 million check-ins, whose repartition can be found in [Table 1](#).

4.1.2 *Venues*

As explained, each of these check-ins is associated with a venue. Thus, the second part of the collection process was to gather information about all of those 340,970 that appear in at least one check-in. This was done by calling the appropriate Foursquare API and the obtained object is described in [Table 11](#) on page 50. The main informations are the venue location, its category, the number of likes, the number of check-ins and the number of unique visitors.

4.2 FLICKR

Flickr is a popular photo sharing service that, as of 2014, has tens of millions of users. It enables users to upload, store, and share their photos, publicly or with their friends. Moreover, it allows users to geo-locate their photos and tag them with descriptive keywords.

Using the Flickr API, we downloaded meta-data from every photo satisfying a set of criteria: they contained at least one tag, they were located inside one of the chosen city and they have been uploaded after January 1st, 2008. This yields 8,358,327 photos but as showed in [Table 1](#), there is a factor of 62 between the figures in New-York and Helsinki.

³ twitter.com

⁴ Using the Public streams API described here: dev.twitter.com/docs/streaming-apis/streams/public.

⁵ a practical description of this dataset can be found online at infolab.tamu.edu/static/users/zhiyuan/icwsm_2011_readme.pdf.

City	Check-ins			Venues	Photos
	2010	2014	Timeline		
New York	408,584	373,005	362,979	67,803	1,744,890
Los Angeles	165,463	141,393	119,364	33,829	794,537
Chicago	133,822	138,642	71,470	26,249	601,859
San Francisco	104,363	78,024	296,050	15,641	983,723
London	72,674	114,008	89,389	23,618	1,653,489
Washington	75,984	81,002	291,466	13,242	650,882
Seattle	51,574	33,034	35,616	9,685	525,794
Amsterdam	35,339	24,676	21,647	8,793	148,807
Houston	41,037	50,642	14,932	13,567	41,827
Atlanta	40,798	43,370	22,797	8,153	228,744
Paris	32,952	68,297	207,895	17,338	208,895
Stockholm	10,501	8,517	70,377	4,204	37,422
Indianapolis	30,955	28,006	9,855	7,709	30,908
Moscow	17,577	334,472	74,129	48,278	69,826
Barcelona	21,448	45,170	161,122	11,444	130,039
Berlin	15,098	40,259	120,151	11,249	226,420
St. Louis	17,491	15,548	7,044	3,482	33,917
Rome	9,364	24,409	74,714	7,725	166,537
Prague	4,757	19,010	8,727	5,748	51,962
Helsinki	6,724	8,394	27,503	3,213	27,849
Total	1,296,505	1,669,878	2,087,227	340,970	8,358,327

Table 1: Number of check-ins, venues, and photos in each city Note that Four-square usage surged in Moscow between 2010 and 2014 whereas the opposite happened in San Francisco.

More precisely, in addition to tags and location, we know when each photo was taken and uploaded, by which user and what title was given to it (the title was not used except when it contained hashtags, which were converted to tags). Thus a typical data point looks like what is described in [Table 10](#) on page 49.

Because of their casual nature, the data contains some inherent noise:

- While timestamp issued by mobile phones are likely to be correct, as their internal clock is synchronized by internet, this may not always be the case for dedicated cameras. More concerning than the usual drift of low quality clocks is the situation of tourists coming from different timezone. Yet as we could not think of any simple solution to that problem, we just ignored it and carried on.
- To ensure the quality of the localization, we restricted results to photos whose precision is deemed “street level” by Flickr. The problem is that it would cost an extra request to know whether

this location was given by GPS (in which case the camera position is accurate) or by the user at upload time. In the latter case, in addition to the general imprecision of the method, it is ambiguous whether this location refers to the place where the photo was shot or the position of the photo’s subject⁶.

4.3 EXPLORATION

While collecting these data, it was helpful to explore them, in order to discover features that could characterize places and help cluster them into similar groups.

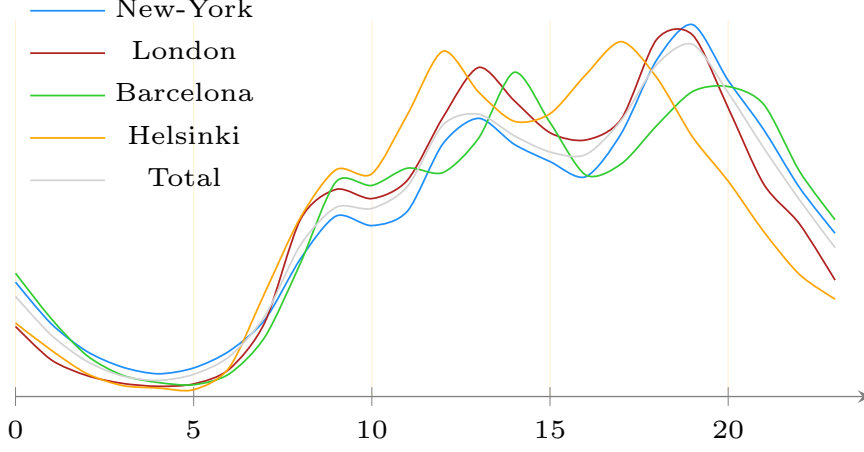
4.3.1 *Time*

The time at which check-ins occur is undoubtedly an important variable. For instance, Kling and Pozdnoukhov [KP12] perform topic modelling from the text of check-in tweets and show that various topics exhibit daily or weekly patterns. Likewise, Zhang, Jin, et al. [Zha+13b] cluster cells of a rectangular grid according to their repartition of check-ins over categories and time of day.

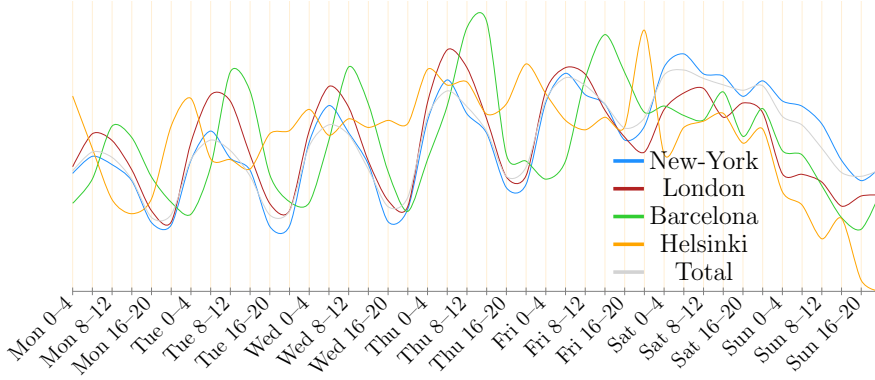
Thus we first look at when check-ins are performed through the day [Figure 2a](#) or the week [Figure 2b](#), and how it differs whether we consider New York, the city that never sleeps, or Helsinki.

But these patterns do not affect all venues uniformly. Taking Paris as an example, we can summarize each venue’s activity by an histogram with 3 or 4 hours bins. Then, running k -means algorithm with $k = 5$, we obtained the clusters depicted in [Figure 3](#). We observe that they have the same characteristic shape with one peak during the day and relatively low activity the rest of the time. We repeat these in all the cities to determine if these clusters are universal. As showed in [Figure 4](#) on page 19, it is the case with 4-hours bins, but 3-hours clusters exhibit slight variations across the cities of our dataset.

⁶ Think of a bridge taken from a nearby hill.



- (a) Hourly check-ins frequency during the day. The activity is at its lowest around 5 a.m. and after that, there are three peaks: one when people go to work in the morning, one in the middle of the day and the last one at the end of the evening. Yet, depending of the city, these peaks do not happen at the same time, nor with the same intensity. Therefore, instead of working directly the raw values of features, we use the number of standard deviation or *z-score*.



- (b) Four-hour check-in frequency during the week. We observe a day/night cycle, which becomes less distinctive as the weekend is approaching and disappears on Saturday.

Figure 2: Check-ins temporal pattern.

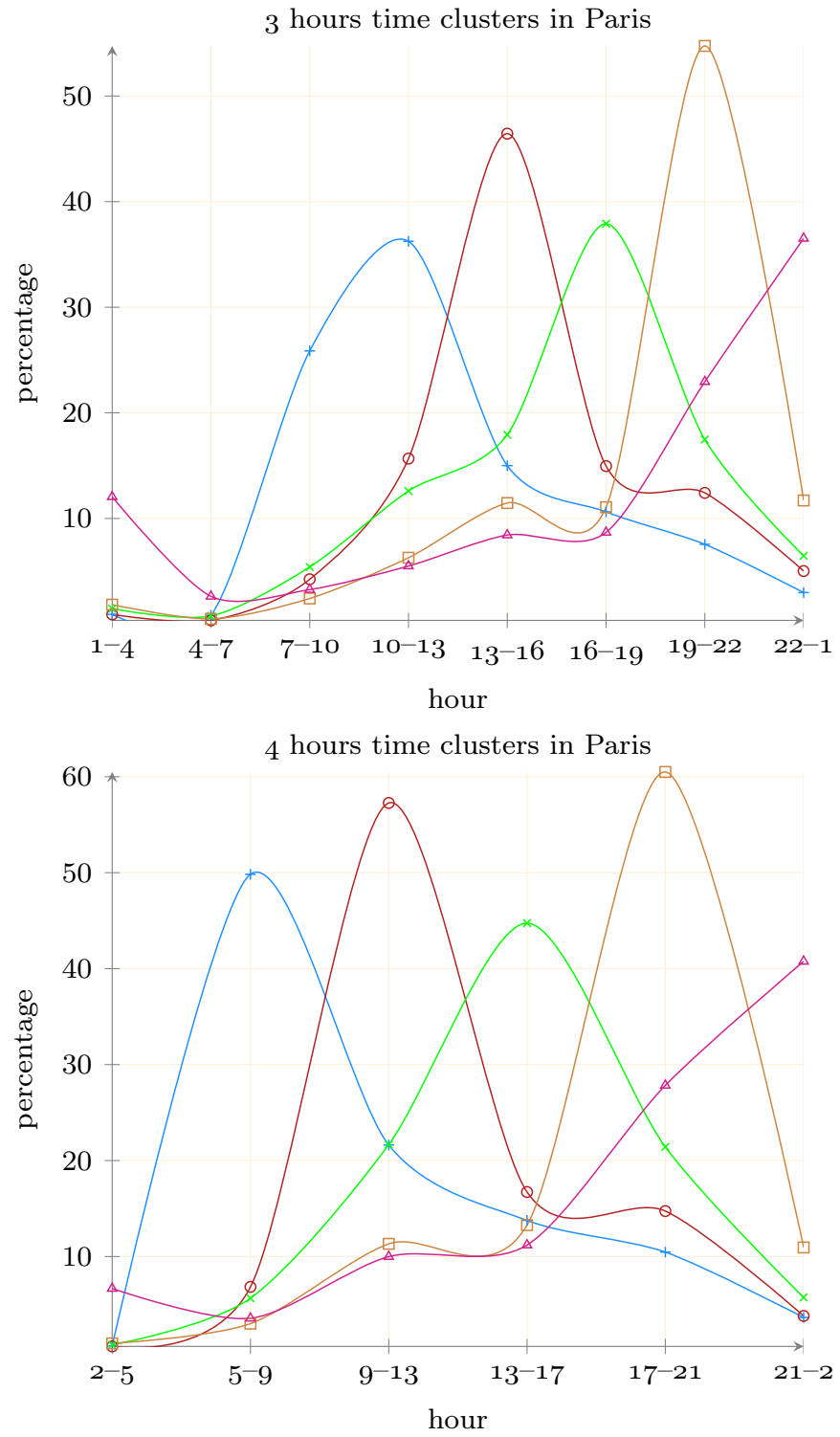


Figure 3: Venues clustered by time of check-ins.

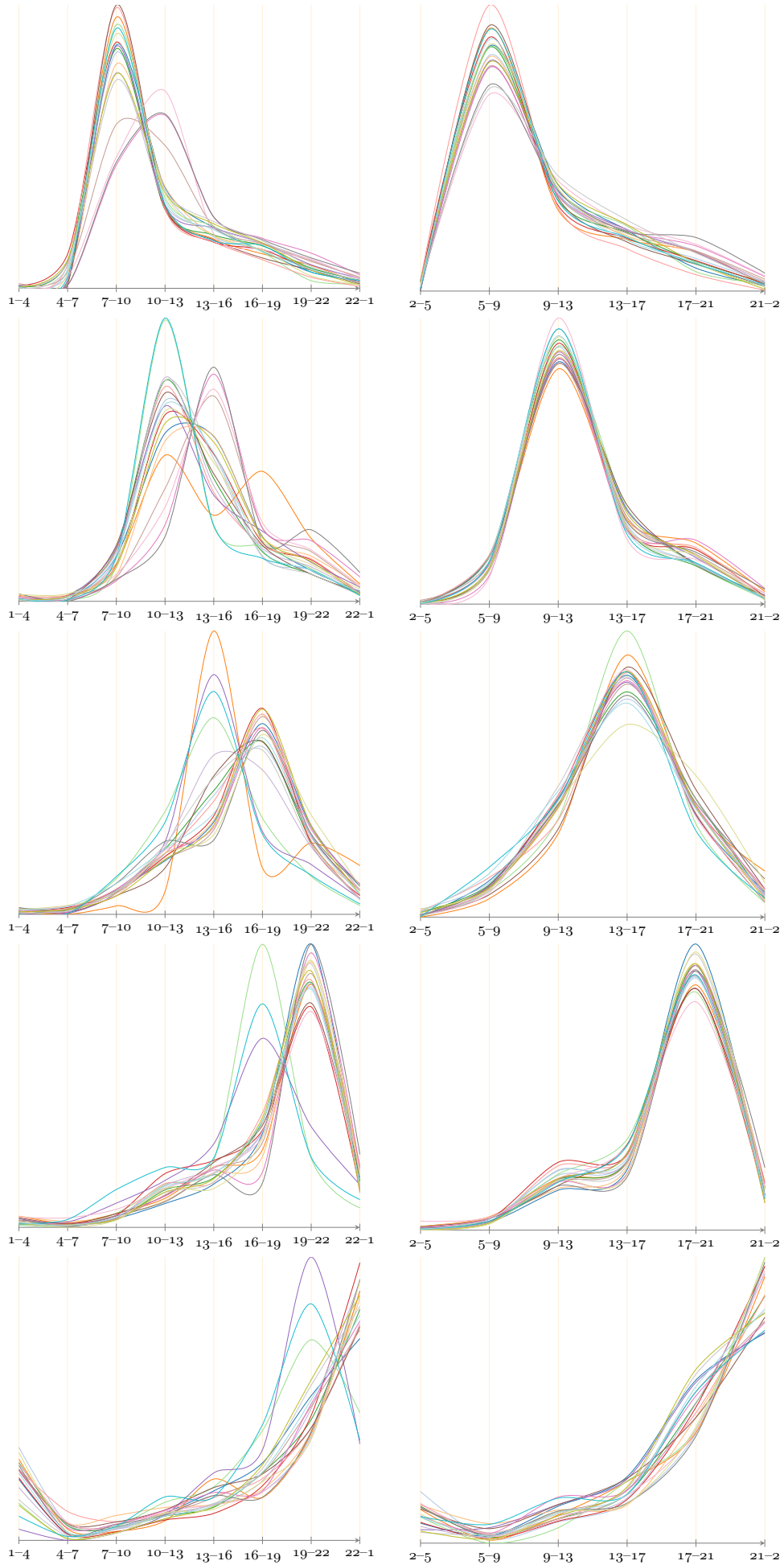


Figure 4: Results of k -means with $k = 5$ on the 3-hours window (left column) and 4-hours window (right column) check-in distribution of venues. Each line in each figure is the centroid of a cluster in a city.

4.3.2 *Space*

The other main characteristic of every check-in is where it takes place. We focus on two spatial feature of venues. First, what does their surrounding looks like in terms of categories repartition. For instance, if a restaurant is mainly surrounded by night life places and another by education buildings, they probably have customers of different kind. Second, venues are not uniformly distributed within the city and the density of nearby locations is a discriminative feature as well. For instance in [Figure 5](#), we can clearly distinguish venues belonging to the city center from the others.

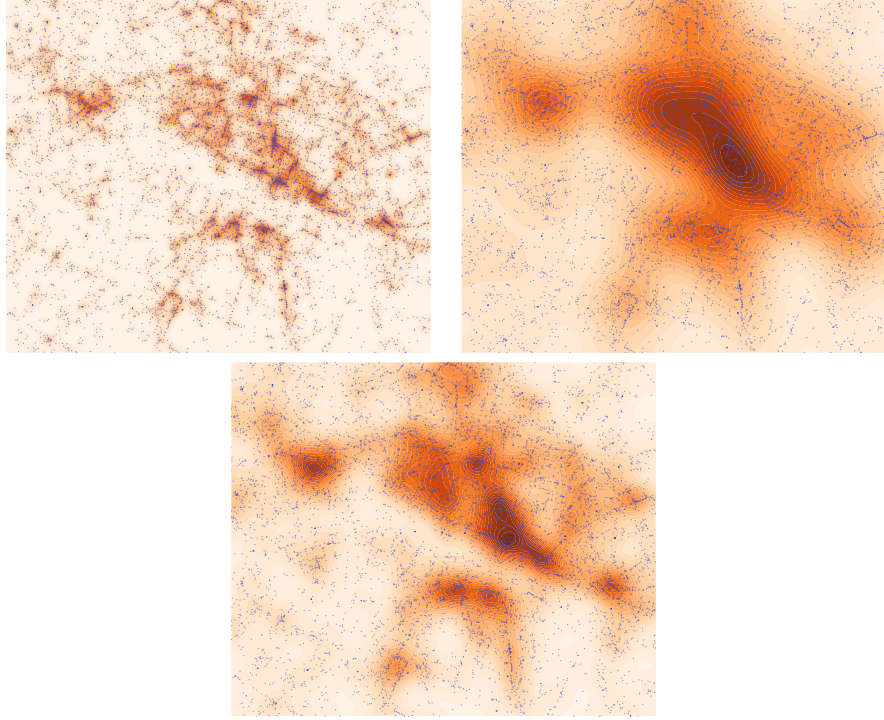


Figure 5: Estimated density of 3192 venues in Paris by a Gaussian kernel, cropped to the center of the city. On the top-left, the bandwidth is 50 meters and it looks too small: venue groups are isolated. In the bottom, $h = 160$ because this value maximizes the probability of existing venues. On the top-right, $h = 350$ corresponds to parameter we choose to define surrounding.

4.3.3 *Location entropy*

Cranshaw, Toch, et al. [[Cra+10](#)] show that “location entropy, which measures the diversity of unique visitors of a location” gives good indication of the sociability level of venues and therefore help predicting friendship ties within [LBSN](#).

Let \mathcal{U} be the set of users and \mathcal{V} the one of venues. For each venue $v \in \mathcal{V}$, we can gather the set of check-ins c_v that occurred there as well as the corresponding users $[u_1, u_2, u_2, \dots, u_n]$ (note that the same users

can check-in multiple times). Then we transform this list to a frequency distribution f_v over \mathcal{U} and compute the normalized entropy:

$$H(v) = -\frac{1}{\log(|\mathcal{U}|)} \sum_{u \in \mathcal{U}} f_v(u) \log(f_v(u))$$

Computing this value for all venues in Paris and Barcelona demonstrates it is a good descriptor of the “publicness” of a place. As showed in Table 6a on the next page, touristic attractions like the Sagrada Família or the Eiffel Tower exhibit high entropy because they are visited by a large and diverse set of people. On the other hand, work offices and private houses have low entropy, as they attract a much more restricted crowd.

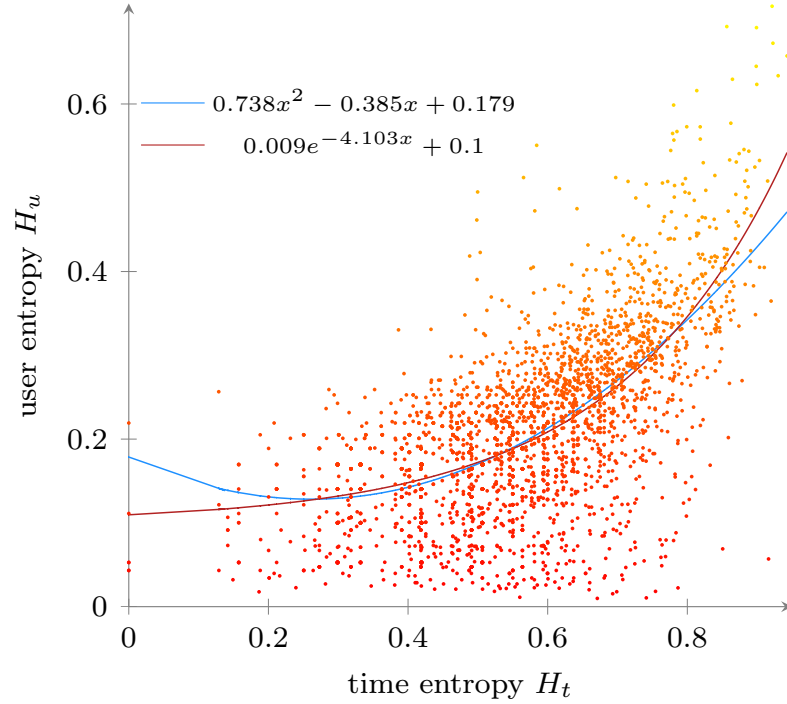
Following the same idea, we also computed the entropy of locations with respect to time of check-ins. Indeed, it distinguishes between a railway station active all day long and a restaurant open only during lunch hours. We also expect a link between the two entropies, as places that stay open longer have more chance to be visited by a various set of users. But this relation, shown on Figure 6b is weak, as some places are visited all the time by a small group of dedicated people.

4.3.4 Photos

Finally, one interesting aspect of this dataset is the interaction between check-ins and photos. The spatial repartition of both objects can be seen as an indirect signal of interest within the city. Thus we were curious to see in which place they disagree, that is location where there are significantly more photos taken than what we would expect by looking at the tweet concentration (and conversely). To answer this question, we compute the discrepancy between these two measures of activity [Aga+06]. It turns out that some places are mostly checked-in because events take place there (such as when people arrive in a new city via a railway station or an airport, or when a sport team plays a match in a stadium) whereas other are more photographed, like museum or touristic sights.

City	Name	Category	Entropy
Barcelona	Castellers de Barcelona	Non-Profit	0.0139
	Café de la Pompeu	Café	0.0172
	Ràdio 4	Radio Station	0.0176
Paris	Boutique Orange	Electronics Store	0.0099
	Métro Goncourt [11]	Subway	0.0105
	Blue Acacia	Office	0.0112
Barcelona	Plaça de Catalunya	Plaza	0.5835
	Sants Estació	Train Station	0.6298
	Sagrada Família	Government Building	0.6309
	Camp Nou	Stadium	0.6852
Paris	Gare SNCF : Gare de Lyon	Train Station	0.6725
	Gare SNCF : Paris Nord	Train Station	0.6911
	Musée du Louvre	Museum	0.6924
	Tour Eiffel	Government Building	0.7167

(a) Venues in Paris and Barcelona with lowest and highest user entropy.



(b) Venues entropy in Paris with respect to user population and time of check-in during the day.

Figure 6: Extreme values of user entropy, and its relation with time entropy.

4.4 CHOSEN REPRESENTATION

Building upon the insights gained by exploring the dataset, we settle to represents each venues by the numeric vector presented in [Table 2](#) on page 26. The features involving the surrounding (numbered from 6 to 16 and 25 to 30) were weighted by a 2D Gaussian of radius $r = 350$ meters.

4.5 HUMAN INPUT

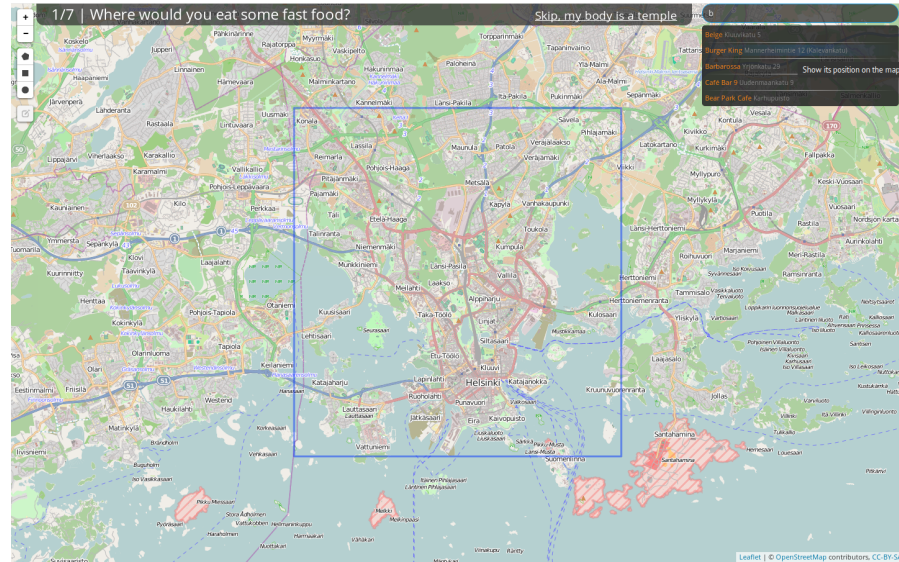
By its nature, the problem is intrinsically unsupervised, for we do not know before hand which place is similar to which one. Furthermore, there is no easily available source containing this information at a large scale. The traditional approach in recommendation system is to held off the most recent data and compare the predicted result with them to assess its accuracy. Yet in this case, this method do not work. First because it restricts test users to those that have spend a significant time in several cities. Second, even among this limited pool, there would still not be enough information to judge similarity. Consider someone who visited ten restaurants in Paris in 2010 and fifteen in London in 2012. We cannot infer from that that they are all similar to each other, which mean we are still facing the original issue, albeit at a smaller scale.

To overcome this difficulty of evaluating results, we decided to ask people for their opinion. But as we thought it was to convoluted to ask directly for pair of locations, we adopted a simpler approach. We presented user with a list of activities (such as “Where would you drink a cozy coffee?”) and a map of a city. For each activity, user are prompted to draw region on the map and optionally, select specific venues. The interface is showed in [Figure 7](#).

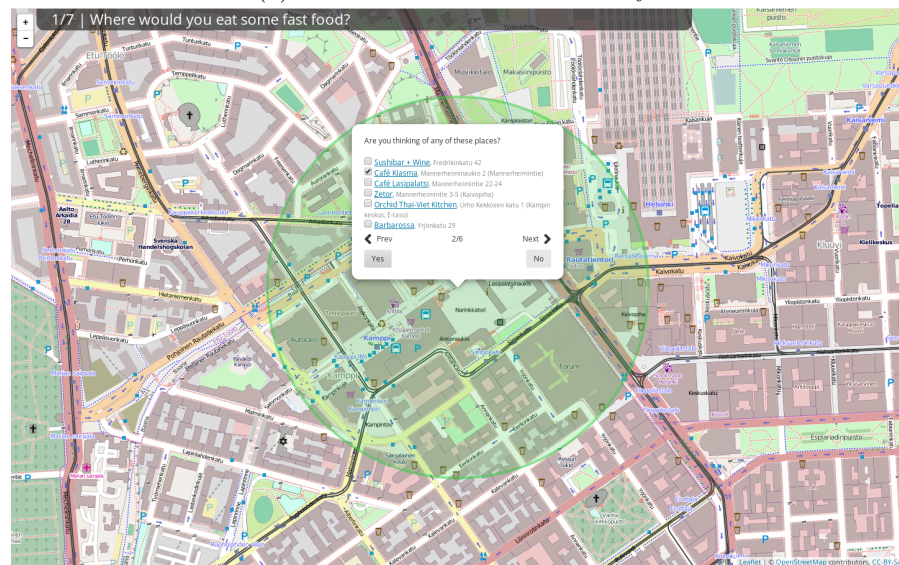
However, we later realized that such information would still be difficult to exploit. Thus we modified the website to collect another kind of data. Namely, we hand picked eight neighborhoods in Paris, presented in [Table 3](#). And we asked international acquaintances to give us similar places in the city where they have lived for some years and that they know well. We obtained answers in Barcelona, New York, Rome, San Francisco, and Washington DC. The answers were slightly altered so that if more than one person provided answers for one city, the answers were merged.

4.6 LIMITATIONS AND OTHER SOURCES

These two datasets suffers from general problem related to online sources. First, as its the case in most of psychological studies [[HHN10](#)], the user distribution of online services is biased toward young people, living in cities and with a high level of life. Second, even among those, we observed an heavy tail phenomena; for instance, 70% of twitter users have tweeted less than 10 times twocharts.com/twitteractivitymonitor.



(a) The main screen of the survey.



(b) After an area have been drawn, the answer can be refined by choosing relevant venues from the dataset.

Figure 7: Website survey interface.

The Foursquare data collection process described above is easily fooled by dishonest users claiming to be at some locations even if it is not the case. According to [Zha+12], this can be caused by the desire to “cheat” the gaming component of many LBSNs, take advantage of monetary incentives offered by venues owner to go in their establishment or simply provide allegedly objective alibi to cover for other activity. Solution involving special hardware and protocol are not within our reach [Pol+13] and even detecting such fake check-ins, by using temporal tensor decomposition [PPF14] is a problem in itself. Therefore we did not make any attempt at cleaning the data.

Another concern is the use of Twitter. First because we access only a sample of all tweets, without any guarantee about its representativeness. Fortunately, it does not appear that Twitter is purposely trying to trick researchers [MPL14]. Second because our tweets span more than three years, a interval during which Twitter and its users behavior have changed [LKM14].

We cannot be oblivious to privacy issues either. Even if it would give us a more reliable picture, there are situations in which people are reluctant (or unable) to check-in [CRH11]. For instance, when people are in hospitals, they have something else on their mind than checking in and as a consequence, these places are not as well described as restaurants or parks.

Finally, through this work, we relied on venues category provided by Foursquare. Yet it has been showed recently than performing Latent Dirichlet Allocation on associated text defined finer and more meaningful categorisation [KIM14], which could bring benefits to the semi supervised training described later.

Colet and Tugores [CT13] describe in great details a setup similar to ours. But there are others ways to obtain information about places. The most straightforward is to ask directly people. For instance, Cranshaw, Luther, et al. [Cra+14] asked residents of Pittsburgh to build a city guide by sharing personal stories about locations they know well. One can also mine text, like Yelp reviews to discover shared topics among restaurant [HRJ14]. Other possibilities include image analysis to describe place atmosphere or mining GPS trajectories to discover which places are significant [CCJ10].

index	description
0	Number of likes
1	Number of unique users
2	Number of total check-ins
3	User entropy
4	Venue density
5	Venue top level category
6	“Arts & Entertainment” venues around
7	“College & University” venues around
8	“Food” venues around
9	“Nightlife Spot” venues around
10	“Outdoors & Recreation” venues around
11	“Shop & Service” venues around
12	“Professional & Other Places” venues around
13	“Residence” venues around
14	“Travel & Transport” venues around
15	Ratio of photos over check-ins
16	Ratio of photos associated to the venue over photos linked to other venues
17	More than half of the check-ins occurs during the week-end
18	Frequency of check-ins between 2 a.m. and 6 a.m.
19	Frequency of check-ins between 6 a.m. and 10 a.m.
20	Frequency of check-ins between 10 a.m. and 2 p.m.
21	Frequency of check-ins between 2 p.m. and 6 p.m.
22	Frequency of check-ins between 6 p.m. and 10 p.m.
23	Frequency of check-ins between 10 p.m. and 2 a.m.
24	Time entropy
25	Frequency of neighbouring check-ins between 2 a.m. and 6 a.m.
26	Frequency of neighbouring check-ins between 6 a.m. and 10 a.m.
27	Frequency of neighbouring check-ins between 10 a.m. and 2 p.m.
28	Frequency of neighbouring check-ins between 2 p.m. and 6 p.m.
29	Frequency of neighbouring check-ins between 6 p.m. and 10 p.m.
30	Frequency of neighbouring check-ins between 10 p.m. and 2 a.m.

Table 2: Feature representation of Foursquare venues, taking surrounding into account. Shaded features are provided directly by Foursquare database whereas the others were computed solely based on the information contained in the dataset.

Name	Description
Golden triangle	Near the Champs-Élysée, it is home of prestigious fashion shops (like Gucci or LVMH) and luxurious palaces (such as Ritz, Georges V or Carlton).
Quartier Latin	A lot of top higher education institutions which draw plenty of reveller students.
Pigalle	The historical red light district.
Montmartre	Touristic spot as well as artsy district.
Official	Here are gathered many official buildings such as the presidential palace, the national parliament and various ministries.
Le Marais	It has welcomed many minorities throughout history, the last one being homosexuals in the 80s.
16 th “arrondissement”	Real estate is expensive there and inhabitants are generally considered to be part of the favored social class.
The banks of the Seine	People can relax there during the weekend with friends or family, close to the nature.

Table 3: List of Paris districts and accompanying descriptions. Participants in the study were asked to identify up to 5 most similar districts in their own city.

5

CHOICE OF METRICS

In this chapter, we describe the various metrics we considered for measuring similarity between venues ([Section 5.1](#)) and neighborhoods ([Section 5.2](#)). We also present which evaluation strategies we devised to choose between several candidates and the final results we obtained.

5.1 VENUES METRICS

As exposed in [Objective 1](#), we want a distance function between two venues, where venues are points in the feature space described in [Table 2](#) on page [26](#).

We evaluate a number of different approaches:

- information theoretic metric learning (ITML);
- gradient boosted large margin nearest neighbor (LMNN);
- embedding to a $2-d$ plane via t -distributed stochastic neighbor embedding (t -SNE) dimensionality reduction method, and computing the Euclidean distance on the projected space;
- Euclidean distance in the original space (EUCLIDEAN).

We described the theory of the first two in [Section 3.2.1](#) but let us provide practical details on how the learning was done in our specific setting. Because these methods are semi supervised, we need some labeling of venues, indicating their similarity. But there is no such labeling easily available. Therefore, we rely on Foursquare top-level category as a proxy, which defines 9 classes. This is problematic for two reasons. First, these categories are so broad that two *Food* venues may not have much in common. Second, the labeling can be noisy: for instance, imagine a museum that most people visit mostly for its cafeteria, and thus it is closer to restaurants than other museums. However, considering the large volume of venues over many cities and with different types, we expect the signal to be stronger than the noise, and the learning algorithms to yield a robust estimation.

Note also that generally, t -SNE is not considered as a metric learning method per se. Yet we already argued that it can learn to uncover relationships between points based on their position in the high-dimensional space. Furthermore, we were struck by the experimental result showed in [Figure 8](#) on the facing page. When we hide category information, venues of the same category are still projected in the same part of the space. Thus, we expect the proximity of venues in this plane to be a good measure of their overall similarity.

As mentioned, there is no similarity ground truth so we evaluate how well these metrics perform on the following tasks:



Figure 8: Using code from [Maa13], we compute the 2D embedding of 23086 venues from 10 cities in Europe. By taking into account all the 31 features but the category of each venue, the method is able to roughly group venues of the same kind: Education on the right, Nightlife on the bottom left, Professional on the bottom right or Recreation on the top.

1. finding venues of the same brand in another city.
2. finding venues of the same category in another city.

5.1.1.1 Brand finding

For this first task, we start by looking at common substrings in the name of venues, keeping those that were company name and not mere generic terms like *park* or *hotel*. We selected *Starbucks* and *McDonald's* as they are ubiquitous and unambiguous.

For each venue v of one of these two brands in one city C , we rank all venues in another city C' based on their distance to v , and consider the highest ranked venue of the same brand. The smaller the rank of this first match, the better the distance measure under consideration. The results for McDonald's are shown in Table 4. LMNN is the best suited measure, even though EUCLIDEAN, without any training, is often close, contrary to t -SNE, which is clearly not up to the task.

Another way to evaluate these measures is to look at the first 540 venues retrieved, and given k , count how many of them before rank k are of this brand (precision) and what proportion of all branded venues are recovered (recall). The average F_1 -score is then given in Table 5 for $k \in \{15, 50, 200\}$ and LMNN again outperforms other measures.

Source	Target	EUCLIDEAN	ITML	LMNN	<i>t</i> -SNE
Berlin	San Francisco	16.9	4.3	3.2	27.9
Chicago	Rome	6.8	7.6	6.5	10.0
Helsinki	London	1.0	1.2	1.2	1.6
Helsinki	Prague	3.0	4.9	1.8	11.2
London	Helsinki	10.0	13.5	7.6	11.1
Moscow	Paris	1.1	1.7	1.5	1.6
New York	St. Louis	15.9	16.4	15.0	25.2
Paris	Moscow	5.6	12.6	3.7	4.3
Prague	Helsinki	6.3	8.2	5.9	16.0
Rome	Chicago	1.5	17.3	1.8	33.9
San Francisco	Berlin	2.8	3.2	2.3	8.9
Seattle	Washington	4.2	2.9	2.2	21.0
St. Louis	New York	5.0	2.0	5.0	33.8
Washington	Seattle	18.3	6.0	12.9	36.0
Winner		3	2	9	0

Table 4: Average percentage of venues in the target city returned before the first McDonald’s.

5.1.2 Category ordering

The second task is more general. Each venue is assigned a hierarchical category by Foursquare¹. We restrict ourselves to two levels, meaning that one venue can be a *French Restaurant* \rightarrow *Food* and another one an *Airport* \rightarrow *Travel & Transport*. We repeat the same sorting procedure but this time, we expect venues of the same sub category to be first, followed by other venues of the same top category and then the rest. To measure how well metrics achieved such ranking, we use **Normalised Discounted Cumulative Gain** [Sak07]. The gain is a measure of relevance. We arbitrarily choose $rel_i = 1$ when the sub category of two venues matches, $rel_i = 0.4$ when only top category matches and $rel_i = 0$ if they were not related at all. We accumulate them (or sum them) but discount results that came too late with

$$DCG = \sum_{i=1} \frac{2^{rel_i} - 1}{\log_2(i + 1)}$$

Finally, we normalized by the result of a optimal ordering to have scores between 0 and 1, which allow direct comparison. The results, shown in Table 6 on page 32, confirm that LMNN and EUCLIDEAN are the best measures regarding this task. It is noteworthy that EUCLIDEAN performs so well without any training.

¹ The complete tree can be seen on their website: developer.foursquare.com/categorytree.

Source	Target	EUCLIDEAN			ITML			LMNN			t -SNE		
Berlin	San Francisco	00	00	08	00	00	10	00	19	17	00	04	07
Chicago	Rome	22	23	19	16	16	19	16	24	22	05	11	13
Helsinki	London	36	20	16	18	25	14	27	25	19	18	15	14
London	Helsinki	28	29	25	24	20	26	34	29	29	38	30	22
Moscow	Paris	15	32	39	08	17	26	08	19	30	08	15	25
New York	St. Louis	06	08	09	12	12	09	10	11	09	04	06	05
Paris	Moscow	03	12	23	03	10	15	04	13	25	03	10	20
Prague	Helsinki	22	36	25	11	32	31	33	36	28	00	12	18
Rome	Chicago	05	13	25	00	06	26	05	06	17	05	06	22
San Francisco	Berlin	12	32	20	06	18	19	23	30	21	06	11	13
Seattle	Washington	09	05	12	09	14	18	35	27	30	09	18	09
St. Louis	New York	00	05	12	06	10	16	06	05	12	00	05	05
Washington	Seattle	04	03	07	04	02	08	12	11	10	04	06	07
Winner		4			3			6			1		

Table 5: 1000 times the F_1 score of McDonald’s at level 15, 50 and 200.

5.2 NEIGHBORHOODS METRICS

We proceed with addressing Objective 2, that is, selecting a distance measure between sets of feature vectors, in order to evaluate similarity between city neighborhoods. First we present the candidates we consider and next how we choose among them.

5.2.1 Candidates

EARTH MOVER’S DISTANCE To compute EMD we assume an underlying metric to measure distances between vectors. We experiment with three variants of EMD: (i) using as the underlying metric the Euclidean distance (EMD-EUCL); (ii) using as the underlying metric the distance learned with LMNN (EMD-LMNN); (iii) using as the underlying metric the Euclidean distance and requiring only a certain fraction of the total mass to be moved (EMD-PARTIAL). The rationale of EMD-PARTIAL is to provide more flexibility by allowing for two neighborhoods to have a fraction of venues that are completely different. For the fraction of the mass required to move, we used 80% in our experiments. Specifically, if we have distribution P and Q with

Source	Target	EUCLIDEAN	ITML	LMNN	<i>t</i> -SNE	Random
Prague	Helsinki	.603	.581	.608	.591	.538
Paris	Moscow	.284	.254	.233	.194	.134
Helsinki	London	.254	.228	.258	.231	.156
Washington	Seattle	.362	.325	.355	.337	.226
New York	St. Louis	.501	.468	.498	.476	.375
San Francisco	Berlin	.376	.341	.374	.351	.246
Rome	Chicago	.222	.187	.224	.219	.140
Paris	Barcelona	.355	.320	.361	.317	.234
Winner		4	0	4	0	0

Table 6: Average NDCG of each measure for a sample of pairs of cities. The last column shows the score obtained by returning venues in a random order, which can be considered as an empirical lower bound on the result.

weights w_P and w_Q summing to one and a ground metric d between them, we find the flow f solving:

$$\begin{aligned}
& \min_f \sum_{i,j} d_{i,j} f_{i,j} \\
& \text{subject to} \quad \sum_j f_{i,j} \leq w_{P,i} \\
& \quad \sum_i f_{i,j} \leq w_{Q,j} \\
& \quad \sum_{i,j} f_{i,j} \geq 0.8
\end{aligned}$$

JENSEN-SHANNON DIVERGENCE JSD can be computed for multivariate distributions, however, in our setting we have a relatively small number of samples (a typical neighborhood contains around 100 venues, and never more than 700) and a large number of dimensions (30 features, see [Section 4.4](#)), so we cannot estimate an accurate joint probability distribution. There are some workarounds [[BDB07](#)] but instead, we opt for computing the JSD independently for each feature, whose density was estimated with a simple 10-bins histogram. In particular, we compute $\text{JSD}_1(F^{(i)}, G^{(i)})$, where JSD_1 is univariate JSD, while $F^{(i)}$ and $G^{(i)}$ are the distributions of the i^{th} feature for the vector sets F and G , respectively. We then aggregate over all features by

$$\text{JSD}(F, G) = \sum_i \theta_i \cdot \text{JSD}_1(F^{(i)}, G^{(i)}). \quad (2)$$

We learn the values of the coefficients θ_i using our collected ground-truth data (described in [Section 4.5](#)). We consider all pairs of neighborhoods obtained and label them as similar if they refer to the same district in Paris (showed in [Table 3](#) on page 27). We then compute θ_i

to maximize the sum of the JSD values over similar pairs minus the sum of JSD values over non similar pairs, subject to the normalization constraint $\sum_i |\theta_i| = 1$.

MINIMUM COST MATCHING DISTANCE A very simple way to compute a distance between two sets of feature vectors is to compute the *centroid* of each set and then compute the distance between the two centroids. We extend this simple definition with k centroids. Given a set of feature vectors, we perform k -means clustering, and represent the set with k centroids. Then, given two sets of feature vectors, and their corresponding k -set centroids, we compute the distance of a min-cost matching, using the Hungarian algorithm. We experiment with different values of k and here we report the best results, obtained for $k = 3$.

All these methods can accept weighted input. We consider using the number of unique visitors for that purpose and thus paying more attention to popular venues but it did not perform better.

5.2.2 Evaluation

5.2.2.1 Methodology

We now describe our evaluation process for selecting the best function for measuring distance over feature vector sets. Consider a neighborhood R in a source city C , and a target city C' . From our user study, we know k ground-truth neighborhoods R_1, \dots, R_k in C' , which are the most similar to R .

Given a distance measure δ we want to evaluate, we can then compute the distance $\delta(F(R), F(R'))$ for each possible neighborhood R' of C' and rank all those neighborhoods in order of increasing distance. We can evaluate the quality of this ranking by checking the position that the ground-truth neighborhoods R_1, \dots, R_k appear in the ranking—if appearing at all. The higher we find a match, the better the ranking, and thus, the better the distance measure δ .

Since we do not have any a-priori neighborhood boundaries (and in fact we do not want to use any, since a neighborhood may be defined dynamically, different from what administrative boundaries would give), any subset of venues that corresponds to a closed and connected region is a candidate neighborhood. As there are exponentially many such subsets, we restrict our search to regions of a certain shape. We consider neighborhoods R' to be *circles* (v', r) , centered at a venue v' and with radius r . We take as v' regularly spaced venues in C' and $r \in \{200, 275, 350, 425, 500\}$ meters, with the additional constraint that the resulting circle should contain at least 20 venues. After ranking all possible such circular neighborhoods R' in order of increasing distance $\delta(F(R), F(R'))$ we remove overlapping neighborhoods.

To evaluate the resulting ranking, we need a *relevance score* for each neighborhood R' in the ranking with respect to the ground-truth neighborhoods R_1, \dots, R_k . Note that R' may not be identical to any of the

ground-truth neighborhoods (for one, R' is circular, while the ground-truth neighborhoods can have arbitrary shapes) but there may have significant overlap with some of them. To account for such overlap, we define the relevance of each R' to be

$$\text{rel}(R' \mid R_1, \dots, R_k) = \max_{i=1}^k \frac{|V(R') \cap V(R_i)|}{|V(R') \cup V(R_i)|},$$

that is, the best *overlap* of R' with a ground-truth neighborhood R_1, \dots, R_k , where the overlap is measured using the *Jaccard coefficient* on the sets of venues of two neighborhoods.

Having assigned a relevance score for each neighborhood R' in the ranking, we evaluate the quality of the ranking using *discounted cumulative gain* (DCG) on the first five candidates.

5.2.2.2 Results

Starting with each of the 6 cities as a source city and for each of the 8 query neighborhoods, we compute the most similar neighborhoods in all other cities, using each of the distance measures that we want to evaluate. The results are shown in [Table 7](#). Each row corresponds to a source city. DCG scores are averaged over all target cities and all 8 query neighborhoods.

We see that EMD-EUCL is the best-performing measure, while JSD performs rather poorly. EMD-LMNN performs only slightly worse than EMD-EUCL. One should contrast this with the results of the previous section, where LMNN slightly outperformed EUCLIDEAN for the task of finding the most similar venue. Thus our results indicate that the nature of the similarity-search task is important for the choice of the distance measure.

Another observation is that the absolute DCG scores of all measures are relatively low. One reason is that many neighborhoods in the ground truth have non-circular shapes (e.g., a long street of luxurious shops). Thus, even if our measures discover an area very close to the ground truth, due to its circular shape, it can have low overlap with the ground truth, and low relevance score. A second unfavorable situation happens when the ground truth contains only one or two small regions. Because we try to find them with five disjoint circles, it must be the case that some of these circles have no overlap at all with ground truth and thus have a relevance of 0. Yet this is merely a comment rather than an issue as here, we are concerned by comparing measures and they all equally suffer from this fact.

	Min cost matching	EMD- EUCL	EMD- LMNN	JSD	EMD- PARTIAL
Barcelona	0.079	0.075	0.080	0.040	0.074
New York	0.056	0.057	0.056	0.054	0.051
Paris	0.061	0.091	0.079	0.045	0.062
Rome	0.020	0.038	0.033	0.018	0.025
San Francisco	0.046	0.045	0.040	0.034	0.044
Washington	0.044	0.035	0.038	0.033	0.039
Average	0.051	0.057	0.054	0.037	0.049

Table 7: Average score of each metric when query are issued from the city on the left. The best metric in each city is **highlighted** and the last row is the average score over all cities.

We now turn our attention to the efficiency aspects of the neighborhood similarity-search problem, i.e., the Problem 1 defined in [Section 2.4](#). The approach used to solve this problem so far is an exhaustive search algorithm, as the one employed in the previous chapter for the evaluation of neighborhood distance measures. Namely, given neighborhood R , consider all candidate neighborhoods R' of a certain shape (circle, rectangle, or other) in the target city C' , evaluate the distance $\text{EMD}(F(R), F(R'))$, and return the neighborhood that achieves the smallest distance.

In this chapter, we present an alternative method to solve the same problem ([Section 6.1](#)). Then we demonstrate experimentally that it accelerates the search task significantly while incurring little loss in accuracy over a wide range of parameters ([Section 6.2](#)).

6.1 METHOD

Our solution relies on the following observation: the EMD between two sets of feature vectors $F(R) = F$ and $F(R') = F'$ is zero, if all feature vectors in F and F' coincide. Relaxing this condition, the EMD is small, if for many vectors in F there is some *near* vector in F' . Put differently, when one feature vector in F is *far away* from all vectors in F' , it contributes a large cost to EMD.

Therefore we can reduce the search space by preprocessing all venues in the target city and keeping only those venues whose feature vectors are close to feature vectors of venues in the query neighborhood. The venues kept in this preprocessing step can be used as *anchors*. We can then look for areas in the target city that are dense in anchor venues, and group them in candidate neighborhoods, for which we calculate the actual EMD.

To see how this idea works, consider [Figure 9](#) on page 38, where we search in Barcelona to find the neighborhood that is most similar to Pigalle in Paris. Each row in the figure corresponds to one venue v in Pigalle, and contains the ranking of all venues in Barcelona sorted by distance to v . There is a cross \times in i^{th} position of the ranking if the i^{th} ranked venue in Barcelona belongs to the ground truth neighborhood. We see that if we restrict ourselves to the 100 nearest neighbors of each venue in the query neighborhood, we recover most of the venues in the ground-truth neighborhood.

We show the effect of varying k in the experiments but first, we try to find a good value a priori. The larger k , the more venues we retrieve, especially the more ground truth venues. But it also makes the search space larger, because it covers a larger proportion of the city. This

trade off is visualized in [Figure 10](#) on the following page, where each point of each query is obtained for a value of k ranging from 1 to 256. We initially chose $k = 50$ because in most cases, it returns 50% of the ground truth venues while covering 33% of the city on average. These numbers hint that the precision is rather low and we need another step to further reduce the search space.

In short, our algorithm works as follows. Starting from the query neighborhood R and target city C' (or cities), we obtain the set of k -nearest neighbors $N_k(v) \subseteq V(C')$ for each venue $v \in V(R)$. All venues found in at least one k -NN set form the set of anchor venues $V_A = \bigcup_{v \in V(R)} N_k(v)$. The set of anchor venues V_A is then treated with respect to its geographic coordinates: the DBSCAN algorithm is applied with initial parameters of $\epsilon = 210$ and $\text{minPts} = 18$. Depending of the results, we recurse over large clusters with stricter parameters or over the whole city with more flexible ones until enough cluster candidates of sensible size are found. Finally, to account for misses that may happen during the pruning phase, each area considered is extended by adding to its boundaries a distance of $j \times \frac{\hat{s}}{\ell}$ meters, where $j = 0, \dots, \ell$ and \hat{s} is the overall size the area. These extended areas are then treated as candidate neighborhoods. At the end of the process, the algorithm returns the neighborhood with the smallest distance (or top- m smallest distances).

Note that although we focus on EMD—because we determine it is the most suited metric to match human judgment—our method is independent of the underlying metric as it plays a role only at the very end of the algorithm.

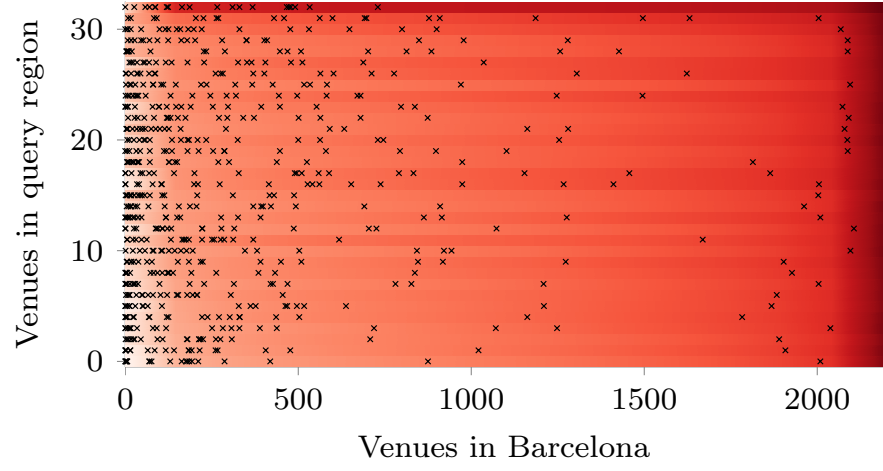


Figure 9: Intuition behind our pruning strategy: for two neighborhoods with small EMD, the venues of one neighborhood are in the k -NN set of the venues of the other.

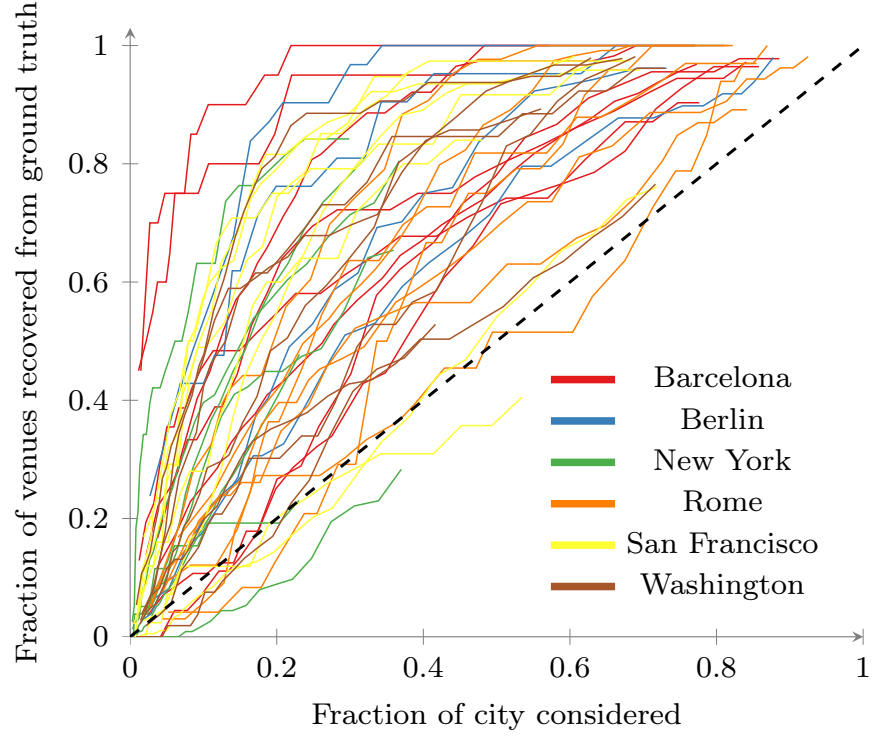


Figure 10: Recall of the k nearest neighbors as a function of the fraction of the total venues in the target city, for k varying between 1 and 256. Each line is query from Paris to another city, indicated by its color.

6.2 RESULTS

6.2.1 How well it approximates exhaustive EMD

Here, we quantify how well our proposed method approximates a brute-force EMD search. We conduct our performance evaluation on the 248 query triplets (C, R, C') that were used in the analysis shown in Table 7.

For each of these queries, we execute the brute-force search, as described in Section 5.2.2; namely, we compute the EMD for all circles (v', r) centered at regularly spaced venues v' of C' and with radius $r \in \{200, 275, 350, 425, 500\}$ meters. We also execute the neighborhood similarity-search algorithm, described above. We compare these two methods in terms of *execution time* and *quality* of solution found. In particular, given a query triple (C, R, C') let R_{BF} be the most similar neighborhood found by the brute force and let R_A be the most similar neighborhood found by the approximation method. Let the corresponding closest distances be $D_{BF} = \text{EMD}(F(R), F(R_{BF}))$ and $D_A = \text{EMD}(F(R), F(R_A))$, respectively. We define the *relative distance* ρ for that query as

$$\rho = \frac{D_A}{D_{BF}}$$

The smaller is ρ , the better the approximation. We would in fact expect ρ to be greater than 1, as values less than 1 indicate that the approximation method is better than the brute force. However, as the brute force is constrained to circular neighborhoods, it is possible that the approximation method finds better solutions. Removing this constraint from the brute force implies searching over other shape families (rectangles, diamonds, etc.), which will increase its running time even more.

Overall, our results show that for the range of parameters we experiment with, the approximation method is much faster than the brute force—in most cases by at least one order of magnitude, while often by two or even three. At the same time the relative distance is very close to 1, often below 1, and rarely above 1.5.

The two parameters of the algorithm, k and ℓ , offer an accuracy vs. efficiency trade-off, that we illustrate empirically. In more detail, we first analyze the effect of ℓ , the number of times we extend the initial regions found after clustering, while using our initial estimate $k = 50$. Intuitively, as ℓ increases, more EMD computations are required, but the chances to find a more similar region increase. Indeed, as we see in Figure 11a on page 41, the relative distance decreases as ℓ increases while the computation becomes more expensive (Figure 11b). We also note that after $\ell = 1$, the gain is small, suggesting that the initial regions are already relevant enough.

We perform the same experiment for $k \in \{8, 25, 50, 80, 160\}$ with $\ell = 1$. As shown in Figure 12a on page 42, the relative distance is very small for all values of k , showing the robustness of the method. At the same time, the running time increases somewhat (Figure 12b), but this

	Approximation	Exhaustive	Biased Approximation
Barcelona	0.066	0.075	0.170
New York	0.029	0.057	0.132
Paris	0.067	0.091	0.191
Rome	0.023	0.038	0.070
San Francisco	0.044	0.045	0.082
Washington	0.071	0.035	0.161
Average	0.050	0.057	0.134

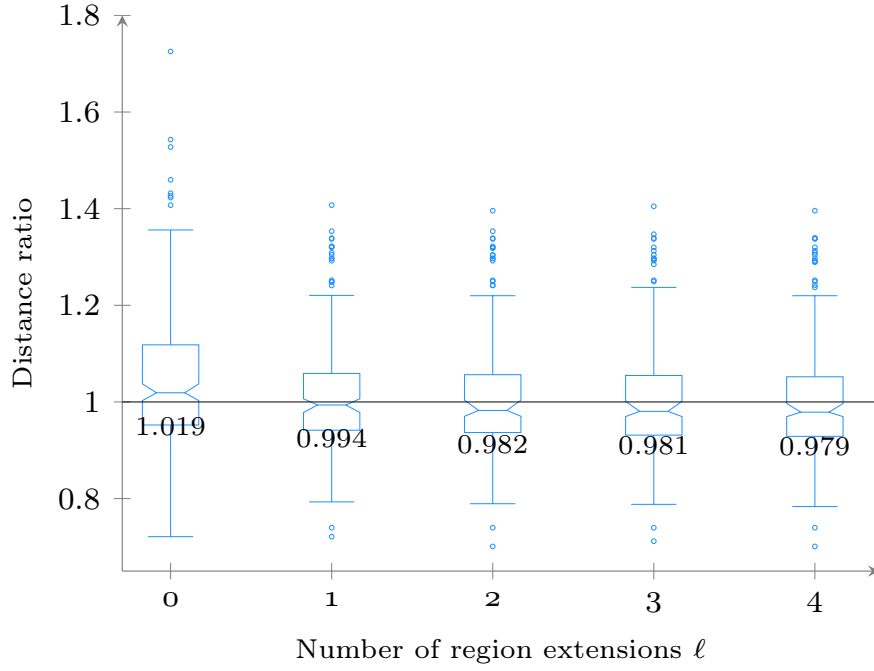
Table 8: Average score when query are issued from the city on the left. The last row is the average score over all cities. The first column is the score obtained when taking the 5 neighborhood with the smallest distance as returned by the approximation method. The second is the result of exhaustive search. In the last one, neighborhoods are ordered to maximize the score.

is due to regions being denser and thus EMD taking more time to reach a solution of the optimization problem.

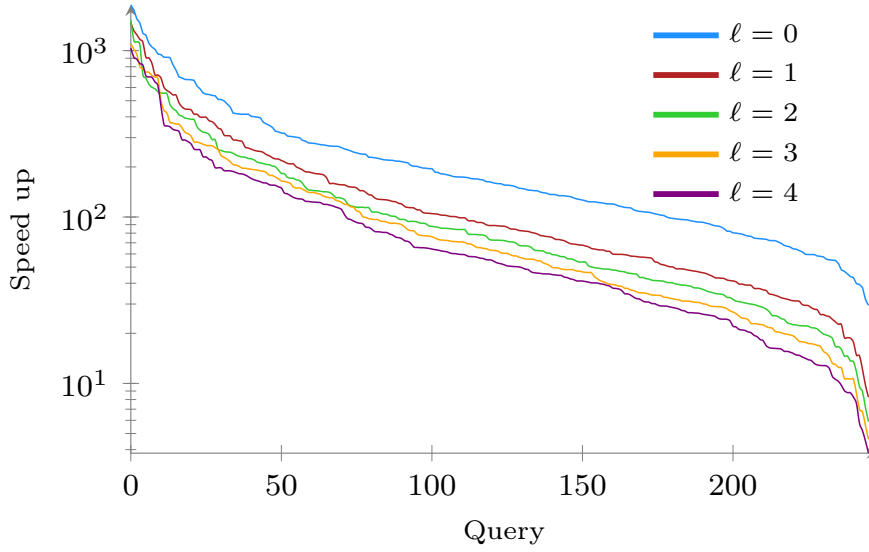
6.2.2 *How well it matches ground truth*

The method we devise finds neighborhoods that generally have comparable distance to the query as those found by exhaustive search, but are obtained much faster. Yet we also want to see if these new neighborhoods retain another characteristics of the exhaustive search: to be close to human ground truth. To answer that, we can perform the same evaluation as in [Section 5.2.2](#). Results showed in [Table 8](#) on the current page indicates that if we sort obtained neighborhood by EMD distance, they perform worse than exhaustive search. Yet the last column shows that among the remaining neighborhoods, some are indeed overlapping with the ground truth. These mixed results can be explained by two problems:

- Sometime, there are too few close neighbors in the ground truth region to form a cluster. Since it was our main assumption, we cannot expect our approximation to return these regions.
- Alternatively, there is a cluster in the ground region but some clusters elsewhere have smaller EMD distance. But this case represents more a failure of EMD to recognize ground truth region and simply indicates that our approximation method cannot alleviate this shortcoming.

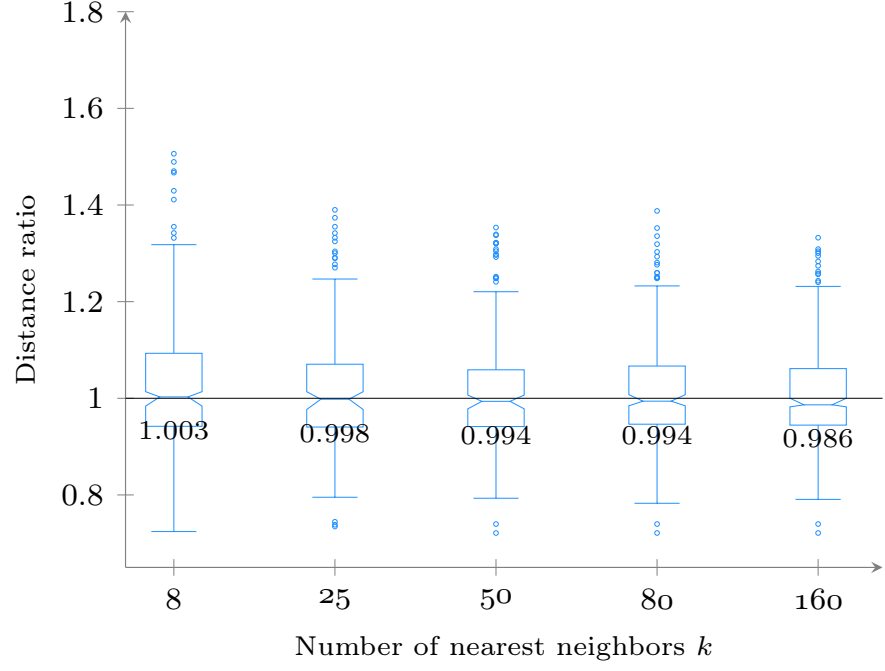
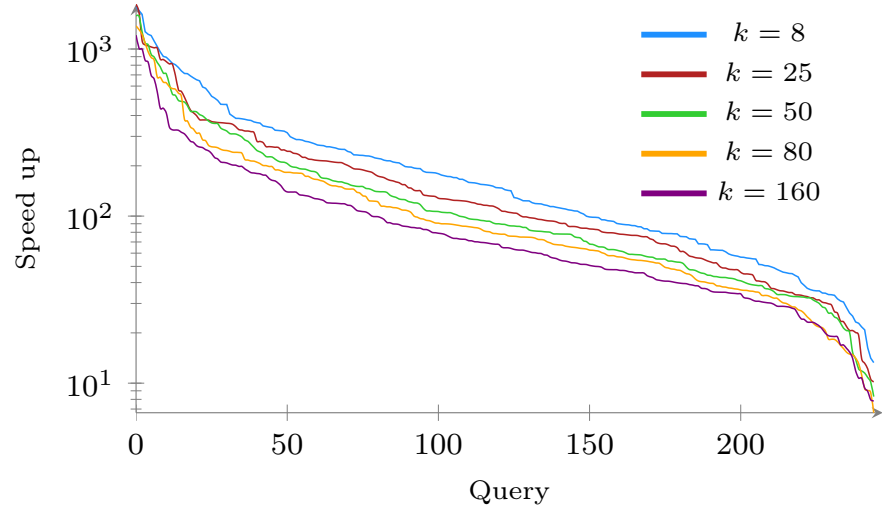


(a) Box plot of the relative distance between nearest neighbor EMD computed by the approximation method and brute-force, as ℓ varies. Values smaller than 1 indicate that the approximation method finds a neighborhood with smaller distance than the brute force.



(b) Time speed-up for each query, in descending order. Note the logarithmic scale.

Figure 11: Approximation trade off while expanding the size of the regions considered.

(a) Same as Figure 11a but showing the effect of varying k .(b) Same as Figure 11b but showing the effect of varying k .Figure 12: Approximation trade off while expanding k , the number of nearest neighbors of each query venues.

RELATED WORK

Our work can be related to several lines of research within the field of urban computing. First, mining spatial data (photos, check-ins, and GPS trajectories) to find places and their semantics, which is a potential application of the broad term of “smart city” and has very concrete implications for major websites. This task also remains of topic modeling, with more constraints regarding space and time. After finding places, we want to learn metrics to compare them in a meaningful and accurate way.

PHOTOS Such places are usually defined by their density, and thus one seeks to find high concentration of people taking photos there. For instance, Deng, Chuang, et al. [DCLog] use DBSCAN to cluster Flickr photos and then look at tags co occurrence within clusters to identify their meaning. Rattenbury and Naaman [RN09] also employ various spatial methods to discover regions in San Francisco where one photo tag appears in burst. The shape of these hotspot regions can be refined by looking at the orientation of each photo [Shi+12]. Another way of identify hotspots is second order Ripley’s K-function [RR12]. Basically, it is a statistical test for the hypothesis that points are distributed according to a Poisson process. It indicates how points “interact with each other, either ‘repulsively’ [...] or ‘attractively’.” Furthermore, it is able to process regions with varying density of pictures. Besides location finding, analyzing photos can also lead to other applications, like recommending routes for tourists based on a graph of points of interest [De +10] and even ensure that these paths are as agreeable as possible [QSA14].

CHECK-INS Moving from Flickr to LBSNs allows searching for larger areas like neighborhoods. Wakamiya, Lee, et al. [WLS12] collect general tweets and perform sentiment analysis as well as movements detection to draw a socio cognitive map of the Kinki region in Japan which helps user to make decision regarding where to live. Closer to our work, Cranshaw, Schwartz, et al. [Cra+12] analyse 18 millions check-ins to find so-called *Livehoods*¹. They build a m spatial neighbors graph of venues with edges weighted by the cosine similarity between their user distribution and then perform spectral clustering. Faced with the same difficulty as ours to evaluate their results, they interview residents of Pittsburgh who validate the obtained subdivisions. Another approach was proposed by Zhang, Noulas, et al. [Zha+13a]², also based on Foursquare check-ins. Each venue is described by its category, its peak time activity and a binary label: touristic or not. They are clus-

¹ livehoods.org

² hoodsquare.org

tered in hotspots along each of these dimensions by the OPTICS algorithm. The city is then divided into a regular grid and cells are described by their hotspot density for each feature. Finally, similar cells are clustered into neighborhoods. The main difference between this last two works and ours is that they do not explicitly mention distance between the neighborhoods they recover.

GPS In addition to static data like photos, another line of work takes advantage of the dynamic nature of human movement to mine trajectories. For instance, Giannotti, Nanni, et al. [Gia+11] analyze GPS data in Italian cities to find temporal patterns, which can then be used for event detection or traffic jam regulation. Similarly, Cao, Cong, et al. [CCJ10] extract stay points from car GPS data and assess their significance by how many visitors go there, how far they traveled to reach them and how long they stay. Uddin, Ravishankar, et al. [URT11] describe efficient techniques to perform closely related tasks. Once the semantics of locations is known, it is still challenging to find frequent patterns efficiently [Zha+14]. Interested reader will find more information in a recent survey [Par+13].

INDUSTRIAL APPLICATIONS The problem of identifying and characterizing neighborhoods has also been addressed by companies. For instance, research in Flickr has shown that by computing the α -shape of a set of tagged photos (which is a generalisation of its convex hull [EKS83]) one can recover boundaries of neighborhoods³. Likewise, Airbnb, a social lodging renting website, has accumulated a lot of spatial data as well as textual reviews, which can be used to rank cities by hospitality⁴ as well as discovering neighborhoods⁵. However, the methods and details of these commercial systems are not publicly available. Finally, engineers at Foursquare used 1.5 billion check-ins to compare activities in different parts of cities in the US⁶.

SPATIO-TEMPORAL TOPIC MODELING Finding such places can also be thought as trying “to discover the hidden thematic structure in large archives of documents”, which is the definition of probabilistic topic modeling according to Blei [Ble12]. Usually such documents are textual and can be combined in any way to form a theme. Here we do not necessarily have text (although photos tags and check-ins contain some) but we have additional spatio-temporal constraints for neighborhoods (or topic) to be localized. For instance, Yin, Cao, et al. [Yin+11] fix a number K of localized topics to be discovered, as well as a set of N weighted Gaussian spatial regions. Then they apply their Latent Geographical Topic Analysis framework: each region has a topic distribution and each topic is a multinomial distribution over all possible photos tags. The parameters of the model are learned by

³ code.flickr.net/2008/10/30/the-shape-of-alpha

⁴ nerds.airbnb.com/most-hospitable-cities

⁵ airbnb.com/locations

⁶ engineering.foursquare.com/2012/03/08/a

an EM algorithm. By taking user into account, this approach provides recommendations [Kur+13]. The same can be done for localized tweets with a hierarchical model of topics [AHS13]. Recently, precision and running time of such methods were improved by Kling, Kunegis, et al. [Kli+14]. Finally, some topics are localized in time as well as in space (like **earthquake**) and this calls for more specific methods. Abdelhaq, Gertz, et al. [AGS13] extract keywords from twitter in a moving sliding window and keep those that are bursty. Using localized tweets, they estimate the topic spatial distribution but because of sparsity and noise, it requires a regularisation based on a co occurrences graph and identification of anchor keywords. A scalable online approximate solution is described in [Bud+13].

SMART CITIES Smart cities in general are a new area which offers a lot of scientific challenges [13, Chapter 4] as well as potential benefits in terms of sustainability and well being by making use of Information Technologies [Doh13]. For instance, IBM was able to tap into Helsinki open data to provide a compelling narrative of citizens' engagement and promote the World Design Capital event in 2012 [Alv+11].

METRIC LEARNING Some pointers were already given in Section 3.2 but the topic is abundant. For instance, learning a Mahalanobis distance was introduced more than ten years ago [Xin+02]. Later, the optimization problem was given a simpler formulation, which has therefore a lower complexity [SKW11]. Two recent works are more sophisticated and use two-stage statistical approach [Wan+14] and sparse local metrics [SBS14].

8

*The future is already here
it's just not very evenly distributed.*

— William Gibson

DISCUSSION

In this chapter, we offer some concluding remarks regarding other applications of our method ([Section 8.1](#)), the overall work performed ([Section 8.2](#)), and some future directions ([Section 8.3](#)).

8.1 OTHER APPLICATIONS

Notice that although we define our problem in terms of cities and venues, it can be generalized to other settings. All we need is a set of disconnected metric spaces¹, points lying in those spaces and described by some features², and some latent high-level characteristics associated with sets of points³. Here are some examples that can be modeled under this framework:

- The social graph of students in several universities. The metric is a distance in the graph and featured points are students. Picking some of them, we posit that they are close because they share the same degree program or some other demographic. We want to know if the same group exists elsewhere.
- Graph of twitter users in different cities or countries. They are described by the topic distribution of their tweets. We ask whether some community of interest exist in several places in the world.
- The set of all books ever written, grouped by centuries and characterised by their style and main themes. Choosing all pieces written during the Enlightenment, it would tell if there was a similar movement in the twentieth century.
- Arranged marriages: points are children, described by their family characteristics. Spaces are set of families, for instance grouped by regions. In practice, one is usually concerned with individual matches but doing it in batch may have pragmatic or social benefits.

8.2 CONCLUSION

In this thesis, we define the problem of matching neighborhoods across cities and introduce a method to solve it efficiently. We start by crawling

¹ Here two dimensional plane supporting the cities.

² Here the venues described by their activity.

³ For instance here, “nightlife neighborhood” or “tourist hotspot”.

social networks for traces of geo-located human activity in 20 major European and American cities. We collect over 8 million geo-tagged Flickr photos from the last six years and almost 5 million Foursquare check-ins. We conduct an extensive exploration of this dataset to reveal pattern of human activity and better understand the relation between our two different sources of information.

Selecting venues as the elementary constituents of a city, we characterize them by features such as time of activity, diversity of audience, popularity and repartition of surrounding venues. We then evaluate several measures of similarity between venues, several of them based on semi supervised metric learning approaches and conclude that, using Foursquare categories as a proxy, EUCLIDEAN and LMNN are the most suitable. Next, we consider various ways of measuring distance between sets of venues and based on neighborhoods ground truth collected through a user study, we find out that EMD outperforms other candidates.

The exhaustive search used in this selection process is time consuming because it evaluates distance function for many regions. Therefore, we develop a method to prune the search space, by pre-processing venues to keep only those close to query venues and clustering them into promising neighborhoods. It greatly enhances run time while incurring only minor loss of accuracy compared with brute force search.

8.3 FUTURE WORK

The methods developed in this thesis—as well as the results they produce—are already satisfactory. Yet we suggest various ways in which they can be improved. We also show how the original problem can be extended to provide even more insights into cities organisation and usage by inhabitants.

The most straightforward way of improving result quality is to use more data sources to derive more features. It is indeed common knowledge that more data trumps better algorithms [HNP09]. In the introduction, we mention for instance data about transportation, weather, air quality, energy consumption, communication, demographics, etc. Although this will inevitably increase the noise, we expect that it will enable more accurate description of venues and areas, making similarity between them even more meaningful.

One comparing metrics between venues, we used metric learning to tune some of them based on the data and conclude that EUCLIDEAN and LMNN were the most accurate. Therefore, we used them as ground metric for EMD when measuring distance between sets of venues. Yet it is also possible to learn the ground metric at this stage, by taking advantage of labeled data regarding sets of venues (and not only venues), as described by Cuturi and Avis [CA14].

Finally, a nice modification will make the algorithm deliver interpretable results. Beyond a single number, the similarity, or absence thereof, must be motivated. It enables principled comparisons of cities

and more user friendly outputs (as well as justifying why no sensible result were found for some queries). Furthermore, this would address recent concerns about Algorithmic Accountability [Dia14]. For instance, Sweeney [Swe13] showed that in case of online advertisement discrimination, opacity of the algorithms makes it difficult to find and sanction those responsible.

In the current situation, the more immediate way to do it would be to find justification after the matching process (by clustering venues and summarizing the results with some text like “These two regions match because people take a lot of photos, go there on weekdays from 4p.m. to 8p.m. and there are a lot of cultural buildings.” Another source of information is the resulting flow of EMD. By telling how much each venue in one region is close to venues in the matching neighborhood, we can recover some explanations. Yet it would be more satisfactory if the whole process was driven by such considerations instead of being a mere post processing step.

As mentioned in some related works [Cra+12; Zha+13a], having a measure of similarity between venues allows clustering them into dense groups sharing similar characteristics. If we include spatial constraints, it corresponds exactly to the definition of neighborhoods, but generated by data instead of some historic administrative decisions. This also reduces the friction for the users of our system. Instead of having to choose a whole region, simply pointing a single location will generate an appropriate neighborhood for the matching process.

A related problem is to match several (or all) neighborhoods from one city at the same time but without excessive overlapping. That is, given a cover of one city, try to find the best coverage of another city along with a mapping between them. This is more computationally challenging but would allow answering questions such as: how homogeneous are European cities compared with American ones? What is the most European city in the United States? Are there common sub structures shared by all cities worldwide? Conversely, what are the places that are unique to Paris or Helsinki and do not exist anywhere else?

Finally, it would be fruitful to apply the same method to similar problems (such as those mentioned in the the beginning of this chapter) to see how well it generalizes and maybe get new ideas on how to improve it.

DATASET FORMAT

field	description	example
<code>_id</code>	Check-in id and its public signature, which can be used to get more information using Foursquare API.	"533bae24498eea4314f699e1?s=aDw52maNIBKg8_vmChduwSKDm3I"
<code>lid</code>	Foursquare venue id	"4d56bccecff7721ea778b9f5"
<code>uid</code>	Numerical Foursquare user id	9037626
<code>loc</code>	GeoJSON location of the tweet	"type": "Point", "coordinates": [2.3979855, 48.85740119]
<code>city</code>	Corresponding city	"paris"
<code>time</code>	Local time at which the check-in took place (this may differ from the time of the tweet)	"2014-04-02T13:28:52Z"
<code>tid</code>	Tweet id	"451244858809012224"
<code>tuid</code>	Twitter user id	"824282623"
<code>msg</code>	Textual content of the tweet	"Should be forgotten (@ Kingdom of Paradise) http://t.co/2CFVAZL1wq "

Table 9: Description of the fields of the check-in object.

field	description	example
<code>_id</code>	Flickr photo id	12683984675
<code>loc</code>	GeoJSON location of the photo	"type": "Point", "coordinates": [25.009667, 60.227586]
<code>uid</code>	Flickr user id	"40747751@No8"
<code>taken</code>	Local time of the shooting according to the camera	ISODate("2014-02-21T18:40:21Z")
<code>title</code>	User given title	"One Way to Lock a Bicycle"
<code>hint</code>	city name	"helsinki"
<code>tags</code>	Normalized user given tags	["bicycle", "lock", "chain", padlock, "locked"]
<code>venue</code>	Foursquare venue id	null
<code>upload</code>	UTC time at which the photo was uploaded	ISODate("2014-02-22T00:28:32Z")
<code>farm</code>	Used to get photos URL	6
<code>server</code>	Used to get photos URL	"5506"
<code>secret</code>	Used to get photos URL	"74fbd5b60a"

Table 10: Description of the fields of the photo object.

field	description	example
<code>_id</code>	Foursquare venue id	"4ba22d4cf964a5207ce137e3"
<code>name</code>	Name of the venue	"Ravintola Oiva"
<code>loc</code>	GeoJSON location according to Foursquare	"type" : "Point", "coordinates": [24.951011, 60.181716084480286]
<code>cat</code>	id of the primary category	"4bf58dd8d48988d11b941735"
<code>cats</code>	Potential additional categories	["4bf58dd8d48988d1c4941735", "4bf58dd8d48988d143941735"]
<code>checkinsCount</code>	Total number of check-ins in Foursquare	2,556
<code>usersCount</code>	Number of unique user in Foursquare	1,321
<code>tipCount</code>	Number of tip left by users	13
<code>price</code>	Price range, from 1 to 4	2
<code>rating</code>	Rating over 10	7.82
<code>createdAt</code>	Time at which the venue was added to Foursquare	ISODate("2010-03-18T15:40:28Z")
<code>mayor</code>	User id of the mayor at the time of collection	18123276
<code>tags</code>	User contributed tags	["karaoke"]
<code>shortUrl</code>	Short URL appearing in tweets	" http://4sq.com/bV8IRQ "
<code>canonicalUrl</code>	Full URL	" https://foursquare.com/v/ravintola-oiva/4ba22d4cf964a5207ce137e3 "
<code>likes</code>	Total number of like in Foursquare	3
<code>likers</code>	List of at most ten users liking this venue	[14695530, 9936722, 37114401]
<code>city</code>	City name	"helsinki"
<code>closed</code>	True when Foursquare says the venue is closed	null
<code>hours</code>	Always null	null

Table 11: Description of the fields of the venue object.

MORE ACKNOWLEDGMENTS

All this work was done on Linux workstations, using many other open source tool such as L^AT_EX, the Python programming language, the IPython environment [PG07] and many scientific libraries:

- Numpy [WCV11]
- Scipy [J+]
- Matplotlib [Hun07]
- Scikit-learn [Ped+12]
- Pandas [McK10]

The calculations were performed using computer resources within the Aalto University School of Science "Science-IT" project.

BIBLIOGRAPHY

- [AGS13] H. Abdelhaq, M. Gertz, et al. “Spatio-temporal Characteristics of Bursty Words in Twitter Streams.” In: *21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 2013, pp. 194–203. DOI: [10.1145/2525314.2525354](https://doi.org/10.1145/2525314.2525354).
- [Aga+06] D. Agarwal, A. McGregor, et al. “Spatial Scan Statistics: Approximations and Performance Study.” In: *12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*. ACM. 2006, p. 24. DOI: [10.1145/1150402.1150410](https://doi.org/10.1145/1150402.1150410).
- [AHS13] A. Ahmed, L. Hong, et al. “Hierarchical Geographical Modeling of User Locations from Social Media Posts.” In: *22nd International Conference on World Wide Web*. 2013, pp. 25–36.
- [Alv+11] A. B. Alvar, P. Bak, et al. *IBM’s Smarter Cities Challenge: Helsinki*. IBM, 2011.
- [Ank+99] M. Ankerst, M. M. Breunig, et al. “OPTICS: Ordering Points To Identify the Clustering Structure.” In: *1999 ACM SIGMOD International Conference on Management of Data*. 1999, pp. 49–60. DOI: [10.1145/304181.304187](https://doi.org/10.1145/304181.304187).
- [BBS08] M.-F. Balcan, A. Blum, et al. “Improved Guarantees for Learning via Similarity Functions.” In: *21st Annual Conference on Learning Theory*. 2008, pp. 287–298.
- [BHS13] A. Bellet, A. Habrard, et al. *A Survey on Metric Learning for Feature Vectors and Structured Data*. Tech. rep. Laboratoire Hubert Curien UMR 5516, Université de Saint-Etienne, 2013, p. 59.
- [Ben75] J. L. Bentley. “Multidimensional binary search trees used for associative searching.” In: *Communications of the ACM* 18.9 (1975), pp. 509–517. DOI: [10.1145/361002.361007](https://doi.org/10.1145/361002.361007).
- [Ble12] D. M. Blei. “Probabilistic topic models.” In: *Communications of the ACM* 55.4 (2012), p. 77. DOI: [10.1145/2133806.2133826](https://doi.org/10.1145/2133806.2133826).
- [BDB07] S. Boltz, E. Debreuve, et al. “kNN-based high-dimensional Kullback-Leibler distance for tracking.” In: *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '07)*. 2007. DOI: [10.1109/WIAMIS.2007.53](https://doi.org/10.1109/WIAMIS.2007.53).
- [Bud+13] C. Budak, T. Georgiou, et al. “GeoScope: Online Detection of Geo-Correlated Information Trends in Social Networks.” In: *VLDB Endowment* 7.4 (2013), pp. 229–240.
- [CCJ10] X. Cao, G. Cong, et al. “Mining significant semantic locations from GPS data.” In: *VLDB Endowment* 3.1-2 (2010), pp. 1009–1020. DOI: [10.14778/1920841.1920968](https://doi.org/10.14778/1920841.1920968).
- [Che95] Y. Cheng. “Mean shift, mode seeking, and clustering.” In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 17.8 (1995), pp. 790–799. DOI: [10.1109/34.400568](https://doi.org/10.1109/34.400568).

- [Che+11] Z. Cheng, J. Caverlee, et al. "Exploring Millions of Footprints in Location Sharing Services." In: *Fifth International AAAI Conference on Weblogs and Social Media*. 2011.
- [CT13] P. Colet and A. Tugores. "Mining online social networks with Python to study urban mobility." In: *PROC. OF THE 6th EUR. CONF. ON PYTHON IN SCIENCE (EUROSCIPY 2013)*. Euroscipy'13. 2013, pp. 21–28.
- [CRH11] H. Cramer, M. Rost, et al. "Performing a Check-in: Emerging Practices, Norms and 'Conflicts' in Location-sharing Using Foursquare." In: *13th International Conference on Human Computer Interaction with Mobile Devices and Services*. 2011, pp. 57–66. DOI: [10.1145/2037373.2037384](https://doi.org/10.1145/2037373.2037384).
- [Cra+14] J. Cranshaw, K. Luther, et al. "Curated City: Capturing Individual City Guides Through Social Curation." In: *Conference on Human Factors in Computing Systems*. 2014.
- [Cra+12] J. Cranshaw, R. Schwartz, et al. "The Livehoods Project: Utilizing Social Media to Understand the Dynamics of a City." In: *Sixth International Conference on Weblogs and Social Media*. 2012.
- [Cra+10] J. Cranshaw, E. Toch, et al. "Bridging the Gap Between Physical Location and Online Social Networks." In: *12th ACM International Conference on Ubiquitous Computing*. 2010, pp. 119–128. DOI: [10.1145/1864349.1864380](https://doi.org/10.1145/1864349.1864380).
- [Cut13] M. Cuturi. "Sinkhorn Distances: Lightspeed Computation of Optimal Transport." In: *Advances in Neural Information Processing Systems 26*. 2013, pp. 2292–2300.
- [CA14] M. Cuturi and D. Avis. "Ground Metric Learning." In: *Journal of Machine Learning Research* 15 (2014), pp. 533–564.
- [Dav+07] J. V. Davis, B. Kulis, et al. "Information-theoretic Metric Learning." In: *24th International Conference on Machine Learning*. 2007, pp. 209–216. DOI: [10.1145/1273496.1273523](https://doi.org/10.1145/1273496.1273523).
- [De +10] M. De Choudhury, M. Feldman, et al. "Automatic Construction of Travel Itineraries Using Social Breadcrumbs." In: *21st ACM Conference on Hypertext and Hypermedia*. 2010, pp. 35–44. DOI: [10.1145/1810617.1810626](https://doi.org/10.1145/1810617.1810626).
- [De 77] J. De Leeuw. "Applications of convex analysis to multidimensional scaling." In: *Recent Developments in Statistics* (1977), pp. 133–145.
- [DCLog] D.-P. Deng, T.-R. Chuang, et al. "Conceptualization of place via spatial clustering and co-occurrence analysis." In: *2009 International Workshop on Location Based Social Networks - LBSN '09*. 2009, p. 49. DOI: [10.1145/1629890.1629902](https://doi.org/10.1145/1629890.1629902).
- [Dia14] N. Diakopoulos. *Algorithmic Accountability Reporting: On the Investigation of Black Boxes*. Columbia School of Journalism, 2014.
- [Doh13] P. Doherty. *Smart cities: How to build sustainable and resilient environments in an increasingly urbanized world*. In McGraw Hill Financial Global Institute, 2013.
- [DJ94] M.-P. Dubuisson and A. K. Jain. "A modified Hausdorff distance for object matching." In: *12th International Conference on Pattern Recognition*. Vol. 1. 1994, 566–568 vol.1. DOI: [10.1109/ICPR.1994.576361](https://doi.org/10.1109/ICPR.1994.576361).

- [EKS83] H. Edelsbrunner, D. Kirkpatrick, et al. “On the shape of a set of points in the plane.” In: *Information Theory, IEEE Transactions on* 29.4 (1983), pp. 551–559. DOI: [10.1109/TIT.1983.1056714](#).
- [EK72] J. Edmonds and R. M. Karp. “Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems.” In: *Journal of the ACM* 19.2 (1972), pp. 248–264. DOI: [10.1145/321694.321699](#).
- [EI96] A. Efrat and A. Itai. “Improvements on Bottleneck Matching and Related Problems Using Geometry.” In: *Twelfth Annual Symposium on Computational Geometry*. 1996, pp. 301–310. DOI: [10.1145/237218.237399](#).
- [ES03] D. M. Endres and J. E. Schindelin. “A new metric for probability distributions.” In: *IEEE Transactions on Information Theory* 49.7 (2003), pp. 1858–1860. DOI: [10.1109/TIT.2003.813506](#).
- [Est+96] M. Ester, H. P. Kriegel, et al. “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.” In: *Second International Conference on Knowledge Discovery and Data Mining*. 1996, pp. 226–231. DOI: [10.1.1.71.1980](#).
- [12a] *From GPS and Virtual Globes to Spatial Computing - 2020: The Next Transformative Technology*. Computing Community Consortium, 2012.
- [Gia+11] F. Giannotti, M. Nanni, et al. “Unveiling the complexity of human mobility by querying and mining massive trajectory data.” In: *The VLDB Journal* 20.5 (2011), pp. 695–719. DOI: [10.1007/s00778-011-0244-8](#).
- [13] *Global Systems Science and Urban Development*. Tech. rep. March. EUNOIA, 2013.
- [Gut84] A. Guttman. “R-trees.” In: *ACM SIGMOD Record* 14.2 (1984), p. 47. DOI: [10.1145/971697.602266](#).
- [HNP09] A. Halevy, P. Norvig, et al. “The Unreasonable Effectiveness of Data.” In: *Intelligent Systems, IEEE* 24.2 (2009), pp. 8–12. DOI: [10.1109/MIS.2009.36](#).
- [HHN10] J. Henrich, S. J. Heine, et al. “Most people are not WEIRD.” In: *Nature* 466.7302 (2010), p. 29.
- [HR02] G. E. Hinton and S. T. Roweis. “Stochastic neighbor embedding.” In: *Advances in neural information processing systems*. 2002, pp. 833–840.
- [HRJ14] J. Huang, S. Rogers, et al. “Improving Restaurants by Extracting Subtopics from Yelp Reviews.” In: *iConference 2014*. 2014.
- [Hun07] J. D. Hunter. “Matplotlib: A 2D Graphics Environment.” In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: [10.1109/MCSE.2007.55](#).
- [J+] E. Jones, T. Oliphant, et al. *SciPy: Open source scientific tools for Python*.
- [Ked+12] D. Kedem, S. Tyree, et al. “Non-linear Metric Learning.” In: *Advances in Neural Information Processing Systems* 25. 2012, pp. 2573–2581.
- [KIM14] E. Kim, H. Ihm, et al. “Topic-Based Place Semantics Discovered from Microblogging Text Messages.” In: *23rd In-*

- ternational Conference on World Wide Web*. 2014. DOI: [10.1145/2567948.2576955](#).
- [KMK10] S. Kisilevich, F. Mansmann, et al. “P-DBSCAN: A Density Based Clustering Algorithm for Exploration and Analysis of Attractive Areas Using Collections of Geo-tagged Photos.” In: *1st International Conference and Exhibition on Computing for Geospatial Research*. 2010, 38:1–38:4. DOI: [10.1145/1823854.1823897](#).
- [Kli+14] C. C. Kling, J. Kunegis, et al. “Detecting non-gaussian geographical topics in tagged photo collections.” In: *7th ACM international conference on Web search and data mining - WSDM '14*. 2014, pp. 603–612. DOI: [10.1145/2556195.2556218](#).
- [KP12] F. Kling and A. Pozdnoukhov. “When a City Tells a Story: Urban Topic Analysis.” In: *20th International Conference on Advances in Geographic Information Systems*. 2012, pp. 482–485. DOI: [10.1145/2424321.2424395](#).
- [Kur+13] T. Kurashima, T. Iwata, et al. “Geo Topic Model: Joint Modeling of User’s Activity Area and Interests for Location Recommendation.” In: *Sixth ACM International Conference on Web Search and Data Mining*. 2013, pp. 375–384. DOI: [10.1145/2433396.2433444](#).
- [LO07] H. Ling and K. Okada. “An efficient Earth Mover’s Distance algorithm for robust histogram comparison.” In: *IEEE transactions on pattern analysis and machine intelligence* 29:5 (2007), pp. 840–53. DOI: [10.1109/TPAMI.2007.1058](#).
- [LKM14] Y. Liu, C. Kliman-Silver, et al. “The Tweets They are a-Changin’: Evolution of Twitter Users and Behavior.” In: *Eighth International Conference on Weblogs and Social Media*. 2014.
- [LBB05] U. V. Luxburg, O. Bousquet, et al. “Limits of Spectral Clustering.” In: *Advances in Neural Information Processing Systems 17*. 2005, pp. 857–864.
- [MH08] L. V. D. Maaten and G. Hinton. “Visualizing Data using t-SNE.” In: *Journal of Machine Learning ...* 9 (2008), pp. 2579–2605.
- [Maa13] L. van der Maaten. “Barnes-Hut-SNE.” In: (2013).
- [Mac67] J. MacQueen. *Some methods for classification and analysis of multivariate observations*. 1967.
- [Mah36] P. C. Mahalanobis. “On the generalized distance in statistics.” In: *National Institute of Sciences (Calcutta)* 2 (1936), pp. 49–55.
- [McK10] W. McKinney. “Data Structures for Statistical Computing in Python.” In: *9th Python in Science Conference*. 2010, pp. 51–56.
- [MPL14] F. Morstatter, J. Pfeffer, et al. “When is it Biased? Assessing the Representativeness of Twitter’s Streaming API.” In: *ArXiv e-prints* (2014).
- [Mun57] J. Munkres. *Algorithms for the Assignment and Transportation Problems*. 1957. DOI: [10.1137/0105003](#).
- [Omo89] S. M. Omohundro. *Five balltree construction algorithms*. Tech. rep. International Computer Science Institute, University of California at Berkeley, 1989.

- [PLo8] B. Pang and L. Lee. “Opinion Mining and Sentiment Analysis.” In: *Foundations and Trends in Information Retrieval* 2.1–2 (2008), pp. 1–135. DOI: [10.1561/15000000011](#).
- [PPF14] E. Papalexakis, K. Pelechrinis, et al. “Spotting Misbehaviors in Location-based Social Networks using Tensors.” In: *23rd International Conference on World Wide Web*. 2014. DOI: [10.1145/2567948.2576950](#).
- [Par+13] C. Parent, S. Spaccapietra, et al. “Semantic Trajectories Modeling and Analysis.” In: *ACM Comput. Surv.* 45.4 (2013), 42:1–42:32. DOI: [10.1145/2501654.2501656](#).
- [Ped+12] F. Pedregosa, G. Varoquaux, et al. “Scikit-learn: Machine Learning in Python.” In: *Journal of Machine Learning Research* 12 (2012), pp. 2825–2830.
- [PW09] O. Pele and M. Werman. “Fast and robust Earth Mover’s Distances.” In: *2009 IEEE 12th International Conference on Computer Vision*. 2009, pp. 460–467. DOI: [10.1109/ICCV.2009.5459199](#).
- [PG07] F. Perez and B. E. Granger. “IPython: A System for Interactive Scientific Computing.” In: *Computing in Science & Engineering* 9.3 (2007), pp. 21–29. DOI: [10.1109/MCSE.2007.53](#).
- [Pol+13] I. Polakis, S. Volanis, et al. “The Man Who Was There: Validating Check-ins in Location-based Services.” In: *29th Annual Computer Security Applications Conference*. 2013, pp. 19–28. DOI: [10.1145/2523649.2523653](#).
- [QSA14] D. Quercia, R. Schifanella, et al. “The Shortest Path to Happiness: Recommending Beautiful, Quiet, and Happy Routes in the City.” In: (2014), p. 11.
- [RNo9] T. Rattenbury and M. Naaman. “Methods for extracting place semantics from Flickr tags.” In: *ACM Transactions on the Web* 3.1 (2009), pp. 1–30. DOI: [10.1145/1462148.1462149](#).
- [Ros56] M. Rosenblatt. “Remarks on Some Nonparametric Estimates of a Density Function.” In: *The Annals of Mathematical Statistics* 27.3 (1956), pp. 832–837. DOI: [10.1214/aoms/1177728190](#).
- [RTG98] Y. Rubner, C. Tomasi, et al. “A metric for distributions with applications to image databases.” In: *Sixth International Conference on Computer Vision*. 1998, pp. 59–66. DOI: [10.1109/ICCV.1998.710701](#).
- [RR12] M. Ruocco and H. Ramampiaro. “Exploratory Analysis on Heterogeneous Tag-point Patterns for Ranking and Extracting Hot-spot Related Tags.” In: *5th ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. 2012, pp. 16–23. DOI: [10.1145/2442796.2442802](#).
- [Sak07] T. Sakai. “On the reliability of information retrieval metrics based on graded relevance.” In: *Information Processing & Management* 43.2 (2007), pp. 531–548. DOI: [10.1016/j.ipm.2006.07.020](#).
- [SKW11] C. Shen, J. Kim, et al. “A scalable dual approach to semidefinite metric learning.” In: *Computer Vision and Pattern Recognition 2011*. University of Adelaide. 2011, pp. 2601–2608. DOI: [10.1109/CVPR.2011.5995447](#).

- [SBS14] Y. Shi, A. Bellet, et al. “Sparse Compositional Metric Learning.” In: *AAAI14* (2014).
- [Shi+12] M. Shirai, M. Hirota, et al. “Discovering Multiple HotSpots Using Geo-tagged Photographs.” In: *20th International Conference on Advances in Geographic Information Systems*. 2012, pp. 490–493. DOI: [10.1145/2424321.2424397](#).
- [SJo8] S. Shirdhonkar and D. W. Jacobs. “Approximate earth mover’s distance in linear time.” In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 2008, pp. 1–8. DOI: [10.1109/CVPR.2008.4587662](#).
- [Swe13] L. Sweeney. “Discrimination in Online Ad Delivery.” In: *Commun. ACM* 56.5 (2013), pp. 44–54. DOI: [10.1145/2447976.2447990](#).
- [URT11] M. R. Uddin, C. Ravishankar, et al. “Finding Regions of Interest from Trajectory Data.” In: *2011 IEEE 12th International Conference on Mobile Data Management*. Vol. 1. 2011, pp. 39–48. DOI: [10.1109/MDM.2011.12](#).
- [WLS12] S. Wakamiya, R. Lee, et al. “Crowd-sourced Cartography: Measuring Socio-cognitive Distance for Urban Areas Based on Crowd’s Movement.” In: *2012 ACM Conference on Ubiquitous Computing*. 2012, pp. 935–942. DOI: [10.1145/2370216.2370424](#).
- [WCV11] S. van der Walt, S. C. Colbert, et al. “The NumPy Array: A Structure for Efficient Numerical Computation.” In: *Computing in Science & Engineering* 13.2 (2011), pp. 22–30. DOI: [10.1109/MCSE.2011.37](#).
- [Wan+14] J. Wang, K. Sun, et al. “Two-Stage Metric Learning.” In: *ICML 2014*. 2014.
- [WS09] K. Q. Weinberger and L. K. Saul. “Distance Metric Learning for Large Margin Nearest Neighbor Classification.” In: *Journal of Machine Learning Research* 10 (2009), pp. 207–244.
- [12b] *World Urbanization Prospects, the 2011 Revision: Highlights*. Tech. rep. United Nations, Department of Economic and Social Affairs, Population Division, 2012.
- [Xin+02] E. P. Xing, A. Y. Ng, et al. “Distance Metric Learning, with Application to Clustering with Side-Information.” In: *Advances in Neural Information Processing Systems* 15 (2002), pp. 505–512. DOI: [10.1.1.58.3667](#).
- [Yin+11] Z. Yin, L. Cao, et al. “Geographical Topic Discovery and Comparison.” In: *20th International Conference on World Wide Web*. 2011, pp. 247–256. DOI: [10.1145/1963405.1963443](#).
- [Zha+13a] A. X. Zhang, A. Noulas, et al. “Hoodsquare: Modeling and Recommending Neighborhoods in Location-Based Social Networks.” In: *Social Computing (SocialCom), 2013 International Conference on*. 2013, pp. 69–74. DOI: [10.1109/SocialCom.2013.17](#).
- [Zha+14] C. Zhang, J. Han, et al. “Splitter: Mining Fine-Grained Sequential Patterns in Semantic Trajectories.” In: *VLDB Endowment* 7.9 (2014), pp. 769–780.
- [Zha+12] K. Zhang, W. Jeng, et al. “Towards Reliable Spatial Information in LBSNs.” In: *2012 ACM Conference on Ubiqui-*

tous Computing. 2012, pp. 950–955. DOI: [10.1145/2370216.2370426](https://doi.org/10.1145/2370216.2370426).

- [Zha+13b] K. Zhang, Q. Jin, et al. “On the Importance of Temporal Dynamics in Modeling Urban Activity.” In: *2nd ACM SIGKDD International Workshop on Urban Computing*. 2013, 7:1–7:8. DOI: [10.1145/2505821.2505825](https://doi.org/10.1145/2505821.2505825).